AD-A114 343   CONTROL DATA CORP MONTEREY CA   F/G 8/12
PROGRAM MAINTENANCE MANUAL POLAR ICE FORECAST SUBSYSTEM - ARCTI--ETC(U)
OCT 81   P A HARR, T C PHAM, J P WELSH   N00014-81-F-0028

UNCLASSIFIED   NORDA-TN-122   NL

1 OF 2
AD
A114343

12

NORDA Technical Note 122

Naval Ocean Research and
Development Activity
NSTL Station, Mississippi 39529

Program Maintenance Manual
Polar Ice Forecast Subsystem - Arctic

P.A. Harr
T.C. Pham
Control Data Corporation
Monterey, California

J.P. Welsh
Oceanography Division
Ocean Science and Technology Laboratory

October 1981

82 05-10 014

ABSTRACT

This manual provides a complete and detailed description of the dynamic thermodynamic sea ice model (PIFS-N), including all the necessary information for maintenance programmer personnel required to maintain the system.

TABLE OF CONTENTS

SECTION 1 GENERAL DESCRIPTION

1.1 Purpose of the Program Maintenance Manual

The objective of this Program Maintenance Manual for the Dynamic Thermodynamic Sea Ice Model, PIPS-N, is to provide the maintenance programmer personnel with the information necessary to effectively maintain the system.

1.2 Background

Sea ice forecasting programs have been clearly connected with fleet polar operations since the early 1950's. When organizing and conducting the sea lift operations required for the establishment and resupply of arctic bases such as Thule, Greenland and the Distant Early Warning Line across Alaska and Canada, ice reconnaissance and forecasting services were requested.

Since the advent of the underice submarine operations in 1957, ice intelligence prediction services have been requested by submarine commands. Their operational forecast requirements deviated markedly from surface ship or icebreaker requirements. The submarine operator is primarily interested in knowing the distribution, in frequency and size, and depth of ice pressure ridges which may constitute a hazard to underice navigation particularly in shoal water.

The surface ship is primarily interested in knowledge of ice concentration, the distribution of the various stages of ice development and floe sizes.

Escalations of operational activities

in ice-covered waters over the past seven years with attendant navigation problems have focused Navy and national attention on the need for a more reliable sea ice forecasting program.

The recent establishment of the Navy NOAA Joint Ice Center is a strong indication that Navy and other governmental departments are moving forward to meet existing and full operational requirements.

In recent years, important contributions have been made to understand the dynamics of sea ice through the Arctic Ice Dynamics Joint Experiment (AIDJEX). Several mathematical models have been developed. A continuing effort is needed to evaluate the applicability of these models for sea ice forecasting.

The dynamic-thermodynamic Sea Ice Model was developed by the W. D. Hibler III (Hibler, 1979). The Naval Air Systems Command (NAIR-270G) tasked the Naval Ocean Research and Development Activity's Polar Oceanography Branch to implement the model utilizing the Fleet Numerical Oceanography Center (FNOC) environmental data base and Cyber 203.

The model uses atmospheric forecast and analysis data available at FNOC in the operational data base. The model outputs forecasts of ice drift, concentration, thickness, convergence/divergence plus ice and open water growth.

There have been a number of sea ice models appearing in the literature in the recent years. A list of applicable references to this model is presented below:

i) Hibler,W.D., 1980: Modeling a Variable Thickness Sea Ice Cover. Mon. Wea. Rev., 108, 1943-1973.

ii) Hibler, W.D. 1979: A dynamic sea Ice Model. J. Phys. Oceanogr., 9, 815-846.

iii) Pritchard, R.S., 1978: The effect of strength on simulation of sea ice dynamics. Proc. Fourth int. Conf. Port and Ocean Engineerin under Arctic conditions, D.E. Muggeridge, Ed., Memorial University of St. Johns, Newfoundland, 494-505.

iv) Pritchard, R.S., M.D. Coon and M.G. McPhee,1977: Simulation of sea ice dynamics during AIDJEX. J. Prep. Vessel Tech., 99J, 491-497.

v) Thorndike, A.S., and R. Colony, 1980: Large-scale ice motion in the Beaufort Sea during AIDJEX, April 1975 - April 1976. Sea Ice Processes and Models, R.S. Prichard, Ed., University of Washington Press, 249-260.

vi) FNWC User Manual

vi) Computer Operational Manual.

1.3 Terms and Abbreviations

FNOC	Fleet Numerical Oceanography Center
NORDA	Naval Ocean Research and Development Activity
NPOC	Naval Polar Oceanography Center
NEDN	Naval Environmental Data Network

Section 2 SYSTEM DESCRIPTION

2.1 System Application

Fleet Numerical Oceanography Center is a large computer complex, tasked with the mission of providing worldwide environmental support to the U.S. fleets. To accomplish this mission, FNOC collects meteorogical report data, analyzes the data and predicts changes in environmental conditions.

The Dynamic Thermodynamic Sea Ice model developed by Hibler is modified to use FNOC environmental data. The modified model gets input data from the FNCC data base. These data consist of wind data, surface pressure, surface vapor pressure, air specific humidity, air temperature, incoming and outgoing radiation on the FNOC 63 x 63 Northern Hemisphere grid (A01,A10, A11, A12, A16, A16, A20, A21), interpolates these data to the model grid to get initial boundary conditios.

The model computes ice drift, ice concentration, thickness, convergence/divergence rate plus thick and thin ice growth/decay rate.

2.2 Security and Privacy

The sea ice model does not currently have access to classified information nor does it produce classified output. Hence, security should be treated at appropriate levels, and the user is responsible for protection of any material used by the model.

2.3 General Desription

The dynamic thermodynamic sea ice model has been modified to run on the Cyber 203 utilizing the FNOC environmental data base, FNOC library and plotting facility. In general, the modified model can be divided into three modules: -Input Module, Processing Module, Output Module.

Input module creates a model grid on the FNOC 63 x 63 Polar Stereographic grid, sets boundary and initial conditions for the model and obtains and interpolates input data to each model grid point.

The processing module can be divided into two parts which process two different mechanisms: the dynamic mechanism and the thermodynamic mechanism.

The heart of the dynamic part is subroutine RELAX which computes ice drift for the model.

The thermodynamic mechanism is processed by subroutine HEAT which estimates ice thickness, concentration and thick and thin ice growth/decay rate.

The output module calls many FNOC routines to format output data into CRANDIC format, and stores them for plotting and printing. A functional diagram of the model is shown in Figure 1. Figure 2 illustrates more detail of each functional division.

Figure 1

The functional diagram of the
Dynamic Thermodynamic Sea Ice Model

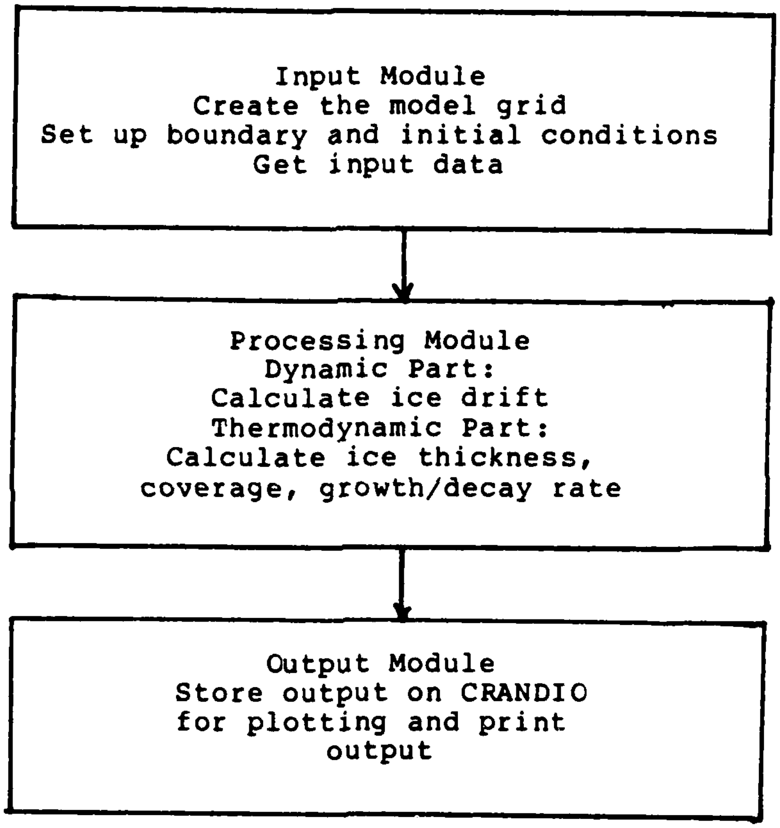
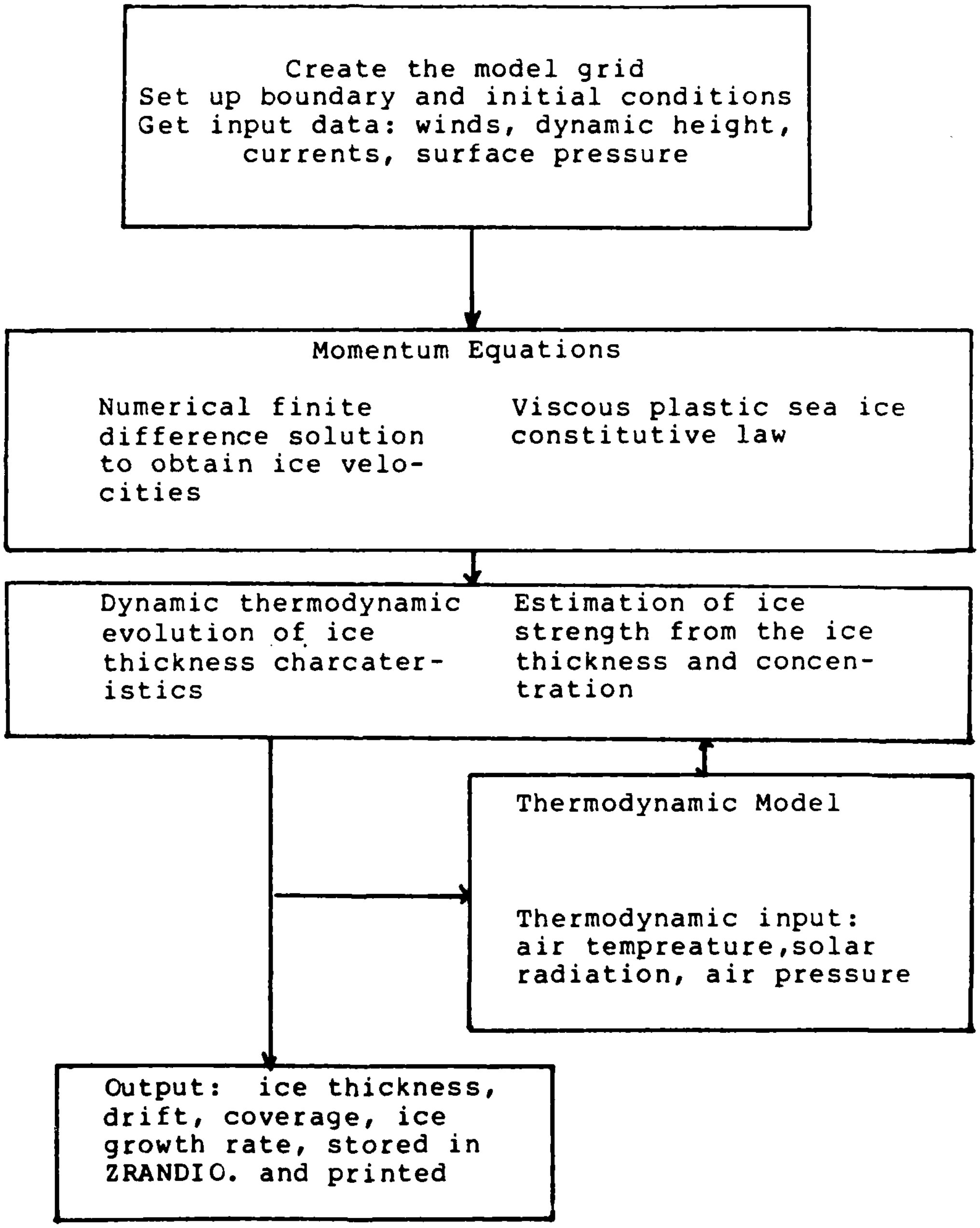


Figure 2

The chart of interrelationship of the major components of the system


Create the model grid
Set up boundary and initial conditions
Get input data: winds, dynamic height, currents, surface pressure
Momentum Equations
Numerical finite difference solution to obtain ice velocities
Viscous plastic sea ice constitutive law
Dynamic thermodynamic evolution of ice thickness charcateristics
Estimation of ice strength from the ice thickness and concentration
Thermodynamic Model
Thermodynamic input: air tempreature,solar radiation, air pressure
Output: ice thickness, drift, coverage, ice growth rate, stored in ZRANDIO. and printed

2.4 Program Description

The following list of subroutine names and functions is provided for easy reference:

NAME	FUNCTION
PROGRAM ICEMDL	Main driving program for the model
	Input Module
SUBROUTINE MESH	Calculates FNOC (I, J) grid points for the model grid;
BNDRY	Sets up boundary masks;
OCEAN	Computes ocean surface currents;
INITIAL	Obtains initial atmospheric data.
	Procesing Module
SUBROUTINE FORM	Sets up forces, drag coefficients, non-linear viscosities for use in each time step;
XSUM	Sums a vector;
PLAST	Calculates non-linear viscosities for plastic flow, used in FORM
UVDDFF	Converts U,V to direction and magnitude and vice versa
INTRP	Performs interpolation from the FNOC grid to the model grid;

RELAX — Solves linearized momentum balance with spatially varying bulk and shear viscosities. Uses sequential over-relaxation technique;

FELLIP
FELLDI — Calculates finite differences for use in RELAX;

ADJUST
MEAN — Estimates thickness and concentration at "open boundary" grid cells based on adjacent values;

ADVECT
DIFFUS — Performs advection of ice thickness and concentration;

HEAT — Driving subroutine for thermodynamic calculations;

BUDGET — Computes thin and thick ice growth/decay rate;

AVG — Computes averaged input variables for staggered grid system;

GROWTH — Calculates changes in concentration and mean ice thickness due to growth and redistribution due to ridging.

Output Module

SUBRCUTINE DIVERG — Output convergence/divergence in CRANDIO format;

UVPLCT — Outputs direction and magnitude of ice drift in CRANDIC format;

HAPLCT — Outputs ice thickness and concentration in CRANDIO format

PRNT — Provides hard copy print out of model variable arrays;

STATPRT — Prints resource use statistics;

GROWPEC — Outputs ice growth/decay rates in CRANDIO format.

2.4.1 Program ICEMDL

A brief description of the theory involved in this model is initially presented in order to help clarify and provide a reference for the subroutine descriptions subsequently presented.

The main driving program, ICEMDL, is designed to call the subroutines which set up the initial conditions, perform the equation integration and output the specified forecasts in CRANDIC format on the CDC CYBER 203 at FNOC.

The overall structure essentially consists of three main components:

i) input and initial conditions;
ii) equation integration;
iii) output processing.

The initial conditions are determined from the atmospheric data available through the FNOC operational data base. A model grid is defined to cover the arctic ocean basin and is a subset of the FNOC hemispheric grid (Figure 1). the grid contains a square mesh with a distance of approximately 120 Km between grid points. Atmospheric variables are interpolated from the FNOC hemispheric grid to the model grid through the use of a 16 point Bessel interpolation scheme. The following atmospheric parameters are interpolated to the model grid;

i) surface wind (u,v);
ii) surface pressure;
iii) surface vapor pressure;
iv) surface air temperature;
v) incoming solar radiation;
vi) total heat flux at the surface;
vii) sensible plus evaporative heat flux at the surface.

Input processing begins by determining the boundary masks to be used in the simulation. The boundary masks define the coastline configuration present within the model grid system. Further output processing involves the incorporation of the initial ocean currents and initial surface wind components. Other input variables listed above are obtained during the processing of the thermodynamic module which accesses the input module to obtain the needed atmospheric thermodynamic variables.

The overall structure of the processing model can also be considered to exist in the three main portions defined below. The first is a momentum balance which includes air and water stresses, coriolos force, internal ice stress and ocean tilt. Non-linear boundary layers for both the air-ice and ocean-ice surfaces are used. A key component is the force due to internal ice stress. This is defined by a constituitive law which relates the ice stress to the strain rate and ice strength. For this model a viscous-plastic constituitive law is followed.

The second feature of the processing module consists of continuity equations describing the evolution of the thickness characteristics on two levels. Two categories of ice thickness are assumed; thin ice (less than 0.5 m in thickness) and thick ice (greater than 0.5 m in thickness). To keep account of these categories two variables are maintained; ice thickness per unit area and the ice concentration which is defined as the fraction of area of a grid cell covered by thick ice. Thermodynamic terms are included on these continuity calculations.

The final component is an ice strength value which is taken to depend linearly upon the ice thickness and exponentially on the ice concentration.

The coupled non-linear equations are treated as an initial value problem using energy conserving finite differences. The momentum equations are integrated implicitly in order to avoid a time limit constraint. A relaxation technique is used on the set of simultaneous equations at each time step.

The numerical scheme uses a staggered grid which allows ice strength and ice velocities to vary in space. To a large degree this staggered grid is patterned after those used in primitive equation ocean models.

As mentioned above, initial conditions at all points and ice velocities at the boundaries are thereafter required to initiate the integration of the system of equations. The most natural condition is to take the ice velocity to be zero at the boundary. This can be accomplished at land boundaries or open ocean boundaries where there is no ice. This boundary condition does not affect the ice motion in such circumstances since, in the absence of ice the strength is zero. It is also possible to set an "open" boundary condition by setting the strength equal to zero near a boundary. This type of open boundary is used at the Spitzbergen-Greenland passage to form a natural inflow/outflow region (Figure 1).

In the computer code, three boundary masks are used to define the "closed" and "open" boundaries. Consequently by altering these masks, highly irregular boundaries may be taken into account.

Because of the strong ice interaction, the momentum equations are parabolic in form and hence have few numerical instability problems over longterm integrations. To avoid non-linear instabilities in longterm simulations which can arise from the non linear advection terms in the continuity equations, small biharmonic and harmonic terms are added to the continuity equation.

A thermodynamic system is incorporated which specifies the ice growth rate as a function of thickness and time of year.

Input to ICEMDL

Input to the ICEMDL program exists on a number of files which must be set up before the actual model is run. Only a portion of these files are actually accessed by PROGRAM ICEMDL. The remainder of the input files is accessed by input module subroutines.

The input files are labeled as TAPE7=IN and TAPE8=DATE in the CY203 convention. TAPE8 contains the date time groups of the period for the model integration. The date time group (DTG) is the standard format defined at FNOC consisting of:

YYMMDDHH
where
YY = year;
MM = month;
DD = day;
HH = hour.

Each DTG is kept in one word and is valid for one day (time step). The DTG is used to access the proper fields from the CRANDIC data base consisting of the atmospheric data maintained by FNOC. Three DTG values are input to ICEMDL at each time step requiring new atmospheric data. The initial DTG values input for the very first time step are set up slightly differently than the other DTG values. For all accesses of the DTG file the first DTG contains the DTG of the current day valid for that time step. The second DTG contains the DTG for 12 hours before the current day. For the initial group of DTG's the third position contains the DTG of the last time step of the previous run. This DTG is used to read the final fields produced by the previous run. These fields are used to restart the model simulation. Therefore, if a run was made to simulate the month of January (ending January 31, 1981) and a new run was to begin for February the initial line of the TAPE8 file would be;

81020100 81013112 81013100.

The DTG, 81020100, is the current DTG for the start of the simulation. The DTG 81013100 is the last day of the previous simulation and will be used to access the data for restarting the model. In reality input atmospheric data is entered into the model every 4 days (time steps). Therefore the DTG values, held in the model, change in increments of 4 days.

The file TAPE7 conatins various types of input data used by the model. TAPE7 is accessed by ICEMDL to set the following variables;

NX, NY - dimension of the grid which holds the momentum variables;

NX1,NY1 - dimension of the grid which holds the thermodynamic variables;

N3 - number of points in the momentum variable grid;

N4 - number of points in the thermodynamic variable grid.

ICEMDL Processing

The processing within ICEMDL begins by accessing the input files described in 2.4.1.1. The next processing involves the accessing of input module subroutines. Subroutine MESH is the first input module routine which is accessed. This routine defines the model grid in terms of the FNOC hemispheric grid. MESH returns the i,j values of the model grid defined in terms of hemispheric grid.

Subroutine BNDRY is accessed after MESH. BNDRY returns the boundary masks to ICEMDL.

Subroutine OCEAN is called to formulate the u, v components of the surface ocean currents and returns them to ICEMDL.

At this point, some basic house keeping tasks are performed such as presetting several arrays to zero and accessing COPEN which is an FNOC routine which "opens" the CRANDIC file to be accessed by the model.

An error condition develops if the routine COPEN finds that it is unable to open the CRANDIC files. The program will terminate, stating that there is a COPEN ERROR on the STOP line of the dayfile.

Subroutine INITIAL is the final input module routine accessed by ICEMDL. Two calls are made to obtain the u and v atmospheric wind components respectively.

The next section of ICEMDL accesses the CRANDIC data base written by previous runs of the model. The variables; ice thickness, ice concentration and ice drift are read to restart the model simulation.

At this point all necessary information is available for the start of processing. The above described code is never executed again during the current run.

The actual equation integration starts with a series of calls to routines FORM and RELAX. FORM computes all parameters necessary for use in the relaxation scheme. These include the air and water drag coefficients, forces terms, ice strength terms, and the viscosity terms. Subroutine RELAX performs the relaxation technique.

There is a sequence of two calls to these routines. The first performs the prediction portion of the momentum time step. Before the first RELAX call,the third level of ice velocities and the centered ice velocities, held in UICE1, VICEC. are set equal to the level of ice velocities by Hibler (1980). During this procedure, the time step is halved. FCRM is called in this first step to use the present ice velocity values to linearize the momentum equations.

The second calls of FCRM and RELAX amount to the main forward time step of the "corrector" section of this "predictor-corrector" method. In this case the "predicted" values of the ice velocities are used in the second FORM call to estimate the viscosity parameters.

After the momentum equations have been implicitly stepped foreward using the relaxation technique, several diagnostic calculations are carried out. The squared ice velocity and the squared ice velocity difference between the times, t and t+1 are computed. This is a simple measure of the change taking place in the ice velocity field during the time step advance.

The predicted ice velocity values are contained in the first level of the UICE and VICE arrays.

The ice velocity and divergence values are passed to the output module routines, UVPLOT and

DIVERG respectively.

Following this momentum time step, the thermodynamic equations are explicitly stepped forward in time. Subroutine ADVECT is called to handle the dynamical portions of the continuity equations for ice thickness and concentration. Subroutine HEAT is called to obtain the thick and thin ice growth rates through the use of a heat budget. Subroutine GRCWTH is called to heat the thermodynamic portion of the continuity equations.

This concludes processing for the time integration. the remainder of the code is used to monitor diagnostic variables and the output modules.

Subroutine XSUM is used to obtain the total ice held within the grid, excluding outflow cells. The following diagnostic values are then computed;

i) Total open water growth for each grid cell; HDIFF;

ii) Net open water growth for the basin; GRSUM1;

iii) Net ice growth for the basin; FHSUM;

iv) Total ice in the outflow cells; TOUT.

The ice held in the outflow cells is explicitly determined through variables, THEFF and THEFF1. The variable THEFF1 contains the amount of ice in the open cells and is computed at the beginning of the time step.

The output module begins with a list of PRINT statements which form the hard copy output. Hard copy output is printed every four time steps. The variable, LSTEP, is used to count time steps and branch to the output section on the fourth time step.

Subroutine PRNT is used to print the model arrays.

Subroutine ADJUST is called after the printed output is formulated. ADJUST is used to define the amount of ice held in the outflow cells for use at the beginning of the next time step.

Subroutine GROWDEC and HAPLOT are used in the output module to output the ice thickness, ice concentrations and growth/decay rates to the CRANDIO file maintained by the model.

The final function performed in ICEMDL is to decide if processing is complete. The variable ITSTEP which is input from TAPE8 defines the total number of time steps minus one to be used in the run. The variable ICOUNT is used to keep track of how many steps have currently been executed.

If more time steps are required, a check is made to determine whether new atmospheric wind data is needed. New wind data is used every four days. the variable, LSTEP, defines the fourth time step as described above. Subroutine INITIAL is called to access the CRANDIO file containing the wind data if needed. The program continues processing until time steps are no longer desired.

If the number of desired time steps has been completed, the diagnostic variables of;

i) outflow;
ii) net ice growth;
iii) net open water growth;

are written to the file TAPE3 for use in a restart run if desired. Subroutine STATPRT is called to print time use statistics on various routines used.

ICEMDL Cutput

The output of ICEMDL consists of forecasts of the following variables;

i) ice thickness;
ii) ice concentration;
iii) ice drift;
iv) ice divergence/convergence;
v) ice strength;
vi) ice growth.

These variables are output in printed form and also in CRANDIC files on the Cy203. CRANDIO is the type of file used operationally at FNOC, on the Cy203, for maintenance of the environmental data base. CRANDIC is analoguous to ZRANDIO on the Cy170's and 6600's at FNOC.

The output CRANDIO files contain one record, produced every four time steps, for each of the above 6 variables. Specific information as to the format and structure of a CRANDIO file can be found in the appropriate FNOC technical write-up.

The records are labeled with the catalog name of each variable. The date of the record, and a tau value. The catalog names are defined as;

i) FFF - ice speed;
ii) DDD - ice direction;
iii) THK - ice thickness;
iv) CON - ice concentration;

v) PRS - ice strength;
vi) HDF - ice growth;
vii) GAR - open water growth;
viii) DIV - convergence/divergence.

Interfaces

Program ICEMDL interfaces with subroutines which comprise the input, processing and output modules. The following table defines all interfaces between these routines and ICEMDL. Specific arrays are defined under Tables and Items.

Subroutine Name			Interfaces
MESH	receives	-	NX, NY, NX1, NY1 NUMBER - number of points in grid;

	returns	-	GRDI - i grid points;
			GRDJ - j grid points;

BNDRY	receives	-	NX, NY, NX1, NY1
	returns		HEFFM - thermodynamic variables boundary mask; UVM - momentum variables boundary mask; OUT - outflow boundary mask;
OCEAN	receives	-	NX, NY, GRDI, GRDJ, UVM;
	returns	-	GWATX - ocean current u components GWATY - ocean current v components;
COPEN	receives	-	IFILE - CRANDIO file name
	returns	-	ISTAT - status of file open attempt;

INITIAL	receives	-	file unit number, NUMBER, GRDI, GRDJ, IDTG, DTG array, ITAU - tau value;
	returns	-	variables read from FNOC data base;

CREADER	receives	-	IFILE - array for data;
			LABEL - record name;
			catalog name;
			record length;

	returns	-	read status, IS;
DDFFUN	receives	-	DD, FF - direction and force of current or drift;
	returns	-	U, V components;
XSUM	receives	-	HEFF - ice thickness array NX1, NY1;
	returns	-	total of array - THEFF or THEFF1;

PRNT	receives	-	array name;
			dimensions of array;
			positions to be printed;

FORM	receives	-	UICE, VICE, ETA, ZETA, AMASS, GAIRX, GAIRY, GWATX, GWATY, OUT, HEFFM, NX, NY, NX1, NY1, HEFF, AREA;
	returns	-	DRAGS, DRAGA, DIV;
RELAX	receives	-	UICE, VICE, ETA, ZETA, AMASS, GAIRX, GAIRY, GWATX, GWATY, DRAGS, DRAGA, OUT, HEFFM, NX, NY, NX1, NY1, HEFF, AREA;
	returns	-	UICE, VICE;
DIVERG	receives	-	DIV, NX, NY, GRDI, GRDJ, IDTG, ITAU;
UVPLOT	receives	-	UICEC, VICEC, GRDI, GRDJ, NX, NY;

ADVECT	receives	-	NICEC, VICEC, HEFF, DIFFI, LAD, HEFFM, NX, NY, NX1, NY1;
	returns	-	HEFF or AREA, DIFFI;
HEAT	receives	-	GRDI, GRDJ, HEFF, AREA, GAIRX, GAIRY, ITAU, IDTG, NX1, NY1, NUMBER;
	returns	-	FC, FHEFF;
GROWTH	receives	-	HEFF, AREA, HC, A22, FHEFF, FO, HCORR, HEFFM, OUT, NX1, NY1;
	returns	-	HEFF, AREA, GAREA;
HAPLOT	receives	-	HEFF, AREA, IDTG, ITAU, GRDI, GRDJ, NX, NY;
GROWDEC	receives	-	HDIFF, FHEFF, GAREA, IDTG, ITAU, NX1, NY1;
ADJUST	receives	-	HEFF, AREA, OUT, HEFFM, NX, NY, NX1, NY1;
	returns	-	HEFF, AREA.

ICEMDL Tables and Items

The following lists define all common blocks and major variable items used in ICEMDL.

I. Common Blocks

/BUOY/ BUOYI, BUOYJ - Buoy position in terms of the FNOC I, J grid;

BX, BY - Buoy positions in terms of the model x, y grid;

/FORCE/ FORCEX - x component of external force plus ice pressure gradient;

FORCEY - y component of external force plus ice pressure gradient;

/STEP/	DELTAT	Time step in seconds;
	DELTAX, DELTAY	mesh size in meters, x, y directions respectively;
/PRESS/	IDENT	CRANDIC record indentification block;
	DATA	hemispheric atmospheric data;
	FILL	filler to put the block on small page boundary (required by CRANDIC software);
/IJ/	IJ63	I,J grid points of SKILES current data;
/DD/	ID	CRANDIC identification block for ice drift direction;
	DD	ice drift directions;
	FILLD	filler for small page boundary;
/FF/	IF, FF, FILLF	Same as /DD/ except for ice drift speeds;
/AR/	IDA, ARRAY, FL	Same as /DD/ except for ice concentration.

Variable Items:

UICE	u component of ice velocity
VICE	v component of ice velocity
ETA	non linear shear viscosity
ZETA	non linear bulk viscosity
GAIRX	u component of wind
GAIRY	v componet of wind
AMASS	ice mass per unit area
HEFF	mean ice thickness per unit area
AREA	fraction of area covered by thick ice
UICEC	Intermediate ice velocities for
VICEC	use in semi-implicit time step
GWATX	u component of ocean currents
GWATY	v component of ocean currents

STRESS	-	XX, YY and XY components of ice stress
FHEFF	-	growth rate of thick ice
FO	-	growth rate of thin ice
DRAGA	-	water drag plus Coriolos parameter
DRAGS	-	water drag plus inertial term
DIV	-	ice convergence/divergence

2.4.2 Input Module Subroutines

2.4.2.1 Subroutine MESH

Subroutine MESH is used to create the model grid. The grid is constructed as a subset of the FNOC hemispheric grid.

Input to MESH

Subroutine MESH receives input from 2 sources. The first source is the formal parameters from ICEMDL. MESH also reads the input file labeled TAPE7. The following variables are contained in the formal parameter list;

i) GRDI, GRDJ - computed in MESH, and contain the i, j coordinates of the model grid

ii) NX, NY, NX1, NY1, N - grid sizes, defined in TAPE 7.

The following variables are obtained from the input file TAPE7;

i) I0, I1 - Defining i grid points on the FNOC hemispheric grid;

ii) J0, J1 - Defining j grid points on the FNOC hemispheric grid;

iii) N - The number of points in each row and column in the model grid.

Subroutine MESH creates a square grid. The variables I0, I1, J0, J1 are defined as follows;

x i, j	x I1, J1
x I0, J0	x i, j

Therefore I0, J0 define the bottom left corner of the desired model grid and I1, J1 define the upper right corner of the model grid. MESH fills in the remainder of the grid, depending upon N. All i, j coordinates are in reference to the hemispheric, 63 x 63 grid of FNOC.

Processing

The processing of subroutine MESH begins by reading all necessary input data from TAPE7. From the variable, N, the values of NX, NY and NX1, NY1 are computed. The actual grid is then formulated using the specified corner points and the number of points desired in each row (column). The column points are defined, stored in GRDJ, followed by the row points stored in GRDI.

After the entire grid is formed, the MESH size is computed utilizing the characteristics and map factor of the polar stereographic grid. The map factor of the model grid is also computed.

The final function of the routine is to compute the initial buoy positions in terms of the newly constructed model grid.

Output

Subroutine MESH produces printed output specifying the following;

i) Corner grid points;
ii) mesh size of model grid;
iii) map factor;
iv) buoy grid locations.

Interfaces

Subroutine MESH does not call any other subroutines.

Tables and Terms

All common blocks and major variables, used in MESH are defined under the ICEMDL section.

2.4.2.2 Subroutine BNDRY

Subroutine BNDRY is called to set up the boundary masks for the thermodynamic, momentum and outflow grids. A boundary mask consists of an array which contains a 1 in grid locations where computations are performed and a 0 on all boundary points. By altering these masks one can obtain any desired boundary or coastline configuration.

Input

The respective boundary masks are read from the input file labeled TAPE7. The grid sizes are input through the formal parameter list interfaced with ICEMDL.

Processing

The standard FORTRAN READ function is used to access TAPE7 and move the boundary masks to the respective locations.

Output

The arrays, UVM, HEFFM and OUT are created by BNDRY.

Interfaces

Subroutine BNDRY does not call any other subroutines.

Tables and Items

The following major variables are used;

UVM - boundary mask for momentum variables;

HEFFM - boundary mask for thermodynamic variables

OUT - boundary mask containing outflow points.

2.4.2.3 Subroutine OCEAN

Subroutine OCEAN is used to define the surface ocean currents. These currents are defined by the SKILES sea ice drift model used at FNOC.

Input

Ocean current directions are read from the input file TAPE7. These direction values are on the model grid. The values were interpolated from the SKILES grid to the current grid and the results placed on TAPE7.

The model grid size and i, j coordinates of the model grid are passed to OCEAN from ICEMDL through the formal parameter list.

Processing

The ocean current magnitudes are located in the DATA statement defining the array WF. The ocean current magnitudes are input from TAPE7 through the use of a standard READ. The current magnitudes are converted from cm/sec to m/sec and placed into a model grid. The current directions, on TAPE7, also needed to be multiplied by 10. For example a direction of 270 is read in as 27.

The current directions and magnitudes are converted to u, v components with the FNOC routine

DDFFUV. Details pertaining to the function of this routine are contained in the standard FNOC subroutine write-ups. The code is placed within this program because the library is not available to the CY203 at the time of this writing.

Output

The arrays GWATX, GWATY are produced in OCEAN.

Interfaces

Subroutine OCEAN interfaces with subroutine DDFFUV which is a standard FNCC library routine used to convert a direction and magnitude to u, v components.

Tables and Items

All major variables contained in OCEAN are defined under ICEMDL.

2.4.2.4 Subroutine INITIAL

Subroutine INITIAL is used to access FNOC CRANDIC data on the CY203 and place the specified atmospheric data into the model grid.

Input

The main input to INITIAL is obtained from ICEMDL through the formal parameter list. The variable, NO, specifies which field is to be accessed from the CRANDIO data base. Value of NO specifies which position in array IRCD is to be used. IRCD holds the catalog names of the various atmospheric data input to the model. The array, IDTG, contains the data time group to be used in reading the data. ITAU is the tau value and N is the number of rows (columns) in the model grid. The arrays GRDI, GRDJ are defined as in MESH.

Processing

The first section of code in INITIAL is setting up the CRANDIO record name. This is a 2 word ASCII label. The first word contains the following;

i) Catalog name of data;
ii) flaps character;
iii) length of record.

The second word contains the DTG. All of this information is masked together in array LABEL. The FNOC routine, CHECKNC, is used to determine whether the specified data is present on the CRANDIC file. If not present the program terminates with the following message;

STOP CHECKNC NO DATA INITIAL.

Providing the data is present, the FNOC routine CREADER is used to read the hemispheric data into array, DATA.

Subroutine INTRP is used to interpolate the hemispheric data to the model grid. INTRP is an FNOC subroutine which performs the interpolating function.

A detailed description of the operations of INTRP is available in the FNOC utility subroutine documentation.

Outputs

Subroutine INITIAL provides for a hard copy print of the requested record label. The array, GAIR, is used to store the resultant data placed in the model grid.

Interfaces

Subroutine INITIAL interfaces with the CRANDIO data base software at FNOC plus a utility library routine, INTRP. Detailed descriptions on all these routines are available as standard products of FNOC.

Tables and Items

The major variables, used in INITIAL, are defined in ICEMDL. The array IRCD is used to maintain the respective catalog names of the CRANDIO data. These catalog names are defined as follows;

i)	A20	-	u wind components;
ii)	A21	-	v wind componments;
iii)	A10	-	atmospheric temperature at the surface;
iv)	A01	-	surface pressure;
v)	A12	-	surface vapor pressure;
vi)	A11	-	shortwave radiation;
vii)	A18	-	total heat flux;
viii)	A16	-	sensible plus evaporative heat flux.

2.4.3 Processing Module

2.4.3.1 Subroutine XSUM

All input to XSUM is provided by ICEMDL, through the parameter list.

Processing

Subroutine XSUM computes a simple sum of an array.

Output

The sum, contained in SI, is produced by XSUM.

Interfaces

No subroutines are called by XSUM.

Tables and Items

No new major variables are defined in XSUM.

2.4.3.2 Subroutine FORM

Subroutine FORM is used to calculate the drag coefficients, external forces and ice strength parameters for use in the time integration of the momentum equations.

Input

Subroutine FORM receives all required input from ICEMDL through the formal parameters. All of the variable definitions passed to FORM are presented under ICEMDL. Certain constants are defined as follows;

i) FCOR - average coriolis parameter;

ii) RHOAIR - air density;

iii) SINWIN - sine of air turning angle;

iv) COSWIN - cosine of air turning angle;

v) SINWAT - sine of ocean turning angle;

vi) COSWAT - cosine of ocean turning angle;

vii) ECCEN - ratio of the axes of the plastic yield curve.

Processing

Subroutine FORM initially calculates the variables required to obtain the external forces operating upon the ice. Within the first major loop the ice mass per unit arrea, coriolis, non-linear water stress coefficient and antisymmetric water drag, DRAGA, are computed. The next major loop calculates the non-linear air stress and the symmetric water drag term, DRAGS. These terms are calculated in a separate loop due to a different grid requirements of the GAIRX, GAIRY (wind component) term (See Appendix A).

The remaining portion of the subroutine deals with finalyzing the external force terms and accessing Subroutine PLAST to compute non-linear shear and bulk viscosities respectively. Before PLAST is called, the ice strength, PRESS, is computed. After PLAST is called the computed viscosities are set to zero at the outflow points by multiplying by the boundary mask OUT.

Finally, the external force components term is combined with the ice pressure gradient.

Output

The results of processing in routine FCRM are stored in DRAGS, DRAGA, FCRCEX, FORCEY, ETA, ZETA, and PRESS. These variables are defined under ICEMDL.

Interfaces

Subroutine FCRM interfaces with routine PLAST. Subroutine PLAST calculates the non linear viscosities based on a Plastic yield curve.

Tables and Items

Subroutine FCRM operates upon a number of major variables which are defined under ICEMDL.

2.4.3.3 Subroutine PLAST

Subroutine PLAST is used by subroutine FORM to calculate the non linear viscosity terms based on a plastic yield curve.

Input

Subroutine PLAST receives all input parameters from FORM, through the formal parameter list.

Processing

The first main loop of PLAST uses the ice drift components to calculate the XX, XY, YY strain rates of the ice (E11, E12, E22 respectively). These components are used, with the constituitive law to calculate the non-linear bulk viscosity. The bulk viscosity is in turn used to compute the non-linear shear viscosity.

The final computation within PLAST is the calculation of the XX, XY, YY components of ice stress plus the divergence is calculated from the XX, YY strain rates.

Output

The main output of PLAST is contained in the arrays ETA, ZETA and sent to FORM through the parameter list.

Interfaces

Subroutine PLAST calls no other subroutines.

Tables and Items

The main variables of PLAST which have not been identified previously are;

i) E11, E12, E22 - XX, XY, YY strain rates respectively.

2.4.3.4 Subroutine RELAX

Subroutine RELAX is the main routine of the processing module. This routine applies a relaxation technique to the dynamical equations for their numerical integration in time. Much numerical detail is contained in the routine and described by Hibler (1979).

Input

Subroutine RELAX receives all input from ICEMDL through the parameter list.

Processing

The processing of RELAX is broken into a number of separate modules. The first 3 major loops perform operations involving the previous time values of u, v plus the evaluation of the diagonal components of the computation matrix.

The main loop (103) performs the iterative relaxation scheme. This loop performs all necessary calculations to obtain a new estimate of the u, v ice drift components (UICE, VICE). After the new components are calculated, 2 checks are made. The first check examines the number of iterations completed to determine if more than 1300 have taken place. If so, the routine shall end printing a message stating more than 1300 iterations have occurred with no convergence. The second check searches for the 100th iteration. At this point the routine switches from a over relaxation scheme to a straight relaxation scheme.

After the checks have been executed, the difference between the new solution and previous iterative solution is computed. If the difference lies within an accepted tolerance the routine ends. If the difference does not meet the tolerance specification another iteration is performed. The old iteration value is stored in the third level of arrays UICE, VICE while the new iterative solution is placed in the first level of arrays.

The relaxation code is made more complex by the separation of the finite difference computations into subroutines, FELLIP and FELLD1. The code is also generalized to fit into the predictor-corrector scheme previously defined. The parameter, THETA, defines whether backwards, centered, or forward time steps are used. A value of 0.5 initiates a centered time and a value of 0 dictates a forward step.

Cutputs

A printed output message is made at the completion of the relaxation procedure. The message states the number of iterations used and the value of the difference between iterations at the end. The results of the processing are placed in UICE, VICE, level 1.

Interfaces

Subroutine RELAX interfaces with routines FELLIP and FELLD1. These routines calculate finite differences used in the relaxation routine.

Tables and Items

A large number of variables are used internally by RELAX. All major variables, however, are defined under the description of ICEMDL.

2.4.3.5 Subroutine FELLIP

Subroutine FELLIP is used to calculate finite differences used in the ralaxation technique.

Input

All required input to FELLIP is provided in the parameter list and passed to FELLIP from RELAX.

Processing

Subroutine FELLIP operates on one grid position at a time. The position is defined by the input variable, i, j, k. FELLIP is called a number of times producing various terms needed by the relaxation routine at each call.

The code remains a very straight forward calculation of the finite difference approximations of the specified terms.

Output

The array, F, holds all resultant calculations.

Interfaces

No further subroutines are called by FELLIP.

Tables and Items

No new major variables are used in FELLIP.

2.4.3.6 Subroutine FELLD1

Subroutine FELLD1 is also used to calculate finite differences for the relaxation code.

Inputs

All required input to FELLD1 is supplied in the parameter list by RELAX.

Processing

Unlike FELLIP, FELLD1 operates upon the entire grid during one call. The level of the array to be altered is specified by input variable, K (e.g., 1, 2, or 3). The input variable, S1, defines the differencing constants.

Outputs

The output of FELLD1 is held within the array, F.

Interfaces

No subroutines are called by FELLD1.

Tables and Items

No new major variables are used in FELLD1.

2.4.3.7 Subroutine ADVECT

Subroutine ADVECT handles the dynamical portions of the thermodynamic continuity equations.

Input

Subroutine ADVECT receives all necessary input from ICEMDL through the formal parameter list. These variables are defined under ICEMDL and any functions performed by these variables are defined below.

Processing

Subroutine ADVECT performs the explicit time stepping procedure of the dynamical portions of the thermodynamic continuity equations. The routine is designed to operate in two separate finite difference schemes. The input variable, LAD, determines whether a backward Euler or Leapfrog scheme is followed. If LAD is 1.0 then the leapfrog scheme is followed, otherwise the Backward Euler is used.

Initially the ice thickness (HEFF) and ice concentrations (AREA) are stepped forward in time by transferring the grid point values to the next lower level. Therefore, the current values are moved to level 2 of the array while the new values are put into level 1. The subroutine is called twice by ICEMDL, one time processing HEFF and secondly processing AREA.

The finite difference approximation to the respective variable and a following check on the finite differencing scheme are the next major processing actions. If the Backward Euler scheme is used (LAD is 0) the scheme is continued to finish the necessary computations required by this scheme.

After the computation is complete for both shemes, the subroutine DIFFUS is called to calculate a smoothing term which is used to comprise the final data value.

Output

The array HEFF, which is internal to ADVECT, holds the final calculations. These values are trasferred to HEFF and AREA in ICEMDL.

Interfaces

Subroutine ADVECT utilizes routine DIFFUS for calculation of a smoothing operator used to reduce the effort of non linear instabilities arising from non linear terms in the continuity equations.

Tables and Items

No major variables are used in ADVECT which have not been previously defined.

2.3.3.8 Subroutine DIFFUS

Subroutine DIFFUS is used to calculate small diffusion terms which are used to reduce instabilities within the non linear continuity equation.

Inputs

All required input for DIFFUS is input to the routine by ADVECT through the formal parameter list. Major input variables are defined under ICEMDL. Any important functions performed by these variables in DIFFUS are detailed below.

Processing

Subroutine DIFFUS applies a simple smoothing operator to obtain a small diffusion term for the respective field being analyzed. This term is applied to the results of the computations of the dynamical portions of the continuity equations.

Output

The diffusion terms are stored in the third level of the array, HEFF.

Interfaces

No major subroutines are accessed by DIFFUS.

Tables and Items

Subroutine DIFFUS introduces no previously defined variables.

2.4.3.9 Subroutine HEAT

Subroutine HEAT is the driving routine for the heat budget code. This budget solves for the thermodynamic growth rate of the thick and thin ice.

Input

Subroutine HEAT receives all input variables from ICEMDL, through the formal parameter list.

Subroutine HEAT also accesses the CRANDIO data base which contains the FNOC environmental data. The following data records are accessed;

i) air temperature;
ii) surface pressure;
iii) surface vapor pressure;
iv) incoming solar radiation;
v) sensible heat flux;
vi) sensible plus evaporative heat flux.

The data is read from the CRANDIC file through the use of subroutine INITIAL of the input module.

Processing

The primary function of subroutine HEAT is to set up all necessary variables for the heat budget calculations, performed in subroutine BUDGET.

The wind data is calculated in the first main loop of HEAT. This variable is calculated from GAIRX and GAIRY which contain the u and v components respectively.

Subroutine INITIAL is called to read the air temperature data at the surface from the CRANDIO data base. Subroutine AVG is called to compute the grid cell average of the air temperature data. Subroutine AVG is used for all thermodynamic variables, within HEAT to define them within the staggered grid. The temperature data is finally converted from centigrade to Kelvin.

Subroutine INITIAL and AVG are again used to define the atmospheric surface pressure and vapor pressure. These variables are used to calculate the moisture at the surface which is stored within QA.

Subroutine INITIAL is finally used to retrieve the shortwave radiation, sensible heat flux and sensible plus evaporative heat flux. The variables are converted from the CGS system to the MKS system and used to calculate the incoming longwave radiation. At the end of the processing the following variables have been set up for use in BUDGET;

i) TAIR - air temperature;
ii) QA - surface moisture;
iii) FSH - incoming shortwave radiation;
iv) FLC - incoming longwave radiation.

The final preparation before BUDGET is called is the definition of the mixed layer depth (HMIX) plus the temperature of the mixed layer and temperature of the ice, TMIX and TICE respectively.

The variable, KOPEN, is used as a flag to BUDGET to determine whether thin ice or thick ice growth rates are evaluated. Subroutine BUDGET is then called to calculate the growth rate of each category. The total growth rate is then calculated and stored in FHEFF.

Output

The environemntal parameters listed under the Tables and Items section are output from HEAT.

Interfaces

Subroutine HEAT interfaces with two main subroutines, INITIAL and BUDGET. Subroutine INITIAL is used to retrieve data from the CRANDIC data base and BUDGET is used to calculate the thin and thick ice growth. The parameter list for INITIAL is defined as follows;

i) parameter 1 - number, indicating which catalog name is to be used in INITIAL;

ii) parameter 2 - array used to store the environemntal data returned from INITIAL;

iii) parameter 3 - i grid points of the grid;

iv) parameter 4 - j grid points of the grid;

v) parameter 5 - DTG required;

vi) parameter 6 - number of points in the grid;

vii) parameter 7 - tau value.

The parameter list of BUDGET is defined as follows;

i) parameter 1 - ice thickness;
ii) parameter 2 - growth rate returned to HEAT;
iii) parameter 3 - flag used to determine whether thin or thick ice growth is calculated;
iv) parameter 4 - grid size;
iv) parameter 5 - grid size;
v) parameter 6 - wind data;
vi) parameter 7 - ice temperature;
vii) parameter 8 - mixed layer temperature;
viii) parameter 9 - air temperature;
ix) parameter 10- surface moisture;
x) parameter 11- incoming longwave.

Tables and Items

The following new major variables are defined in HEAT;

i) TICE - ice temperature;
ii) TMIX - mixed layer temperature;
iii) TAIR - air temperature;
iv) QA - surface moisture;
v) FLO - incoming longwave radiation;
vi) PS - surface pressure and sensible heat flux;
vii) CS - surface vapor pressure and sensible plus evaporative heat flux;
viii) UG - wind values.

The following common block is defined:

/RAD/ - contains incoming shortwave radiation.

2.4.3.10 Subroutine BUDGET

Subroutine BUDGET is used to calculate the thin and thick ice growth rates by using a simple heat budget.

Input

Subroutine BUDGET receives all major input varibales from HEAT. Most of these are passed through the formal parameter list. One variable, FSH, is passed in common block /RAD/.

Processing

The initial code of BUDGET is dedicated to defining all necessary constants used in the heat budget calculations.

A branch is made, depending upon the value of KOPEN. If KOPEN is less than zero, processing continues to compute the thin ice growth rate. If KOPEN is greater than zero processing jumps to compute the thick ice growth rate.

Continuing with the the ice growth, the main heat budget equation components, and growth rate derived within loop 101. Subroutine BUDGET ends processing here for this branch.

When the thick ice growth rate is computed, there are two main components of the heat budget calculation. The ice temperature. TICE, is solved for iteratively (Newton-Raplson technique) and used in the head budget equation. When the ice temperature values are relatively stable between iterations, processing finishes returning the thick ice growth rate.

Output

The output of BUDGET is contained in the following variables;

i) FHEFF - thick ice growth rate;
ii) FO - thin ice growth rate.

Interfaces

Subroutine BUDGET calls no other subroutines.

Tables and Items

The variable, ALB, defining the surface albedo, is the only major variable not previously defined and used within BUDGET.

2.4.3.11 Subroutine GROWTH

Subroutine GROWTH is used to calculate the change in ice thickness amd concentration due to growth/decay.

Input

All required input variables to GROWTH are passed from ICEMDL through the formal parameter list. No variables, not previously defined, are input to GROWTH.

Processing

The processing of GROWTH is contained in one major loop. The amount of ice grown and melted during the time step. The changes, reflected by the growth/decay rates, are added to the ice thickness, HEFF, and ice concentration, AREA, variables. The ice concentration value is then checked to contain it within the limits specified. Ice concentrations are not allowed to be larger than 1.0.

Output

The variables HEFF, AREA and GAREA contain output variables of GROWTH.

Interfaces

No other subroutines are called by GROWTH.

Tables and Items

No new variables are introduced by GROWTH.

2.4.3.12 Subroutine ADJUST

Subroutine ADJUST is used to set up the thickness and concentration at the outflow cells.

Input

All input variables, to ADJUST are passed from ICEMDL through the parameter list. No variables not previously defined are used.

Processing

Subroutine ADJUST is called at the end of each time step. ADJUST uses routine MEAN to calculate the ice in the open cells by taking an average of all grid cells adjacent to the open boundary. The ice thickness and concentration arrays are both modified in this manner.

Outputs

The arrays HEFF and AREA (ice thickness and ice concentration) are respectively modified.

Interfaces

Subroutine ADJUST calls subroutine MEAN to calculate the ice in open grid cells.

Tables and Items

No major new variables are defined.

2.4.3.13 Subroutine MEAN

Subroutine MEAN is used to calculate the amount of ice in open cells.

Input

All data input to MEAN is passed by ADJUST through the parameter list. No new variables are introduced as input.

Processing

Subroutine MEAN calculates the amount of ice in a grid cell as the mean of ice in the adjacent cells. Array CUT, which is the boundary mask specifying the outflow cells, is used to control the calculations to output the mean ice held within the outflow cells.

Cutput

The variable, HMEAN, is output to ADJUST, containing the mean ice thickness and concentration on the open cells.

Interfaces

No other subroutines are called by MEAN.

Tables and Items

No new major variables are introduced by MEAN.

2.4.4 Output Module

2.4.4.1 Subroutine DIVERG

Subroutine DIVERG is used to output the divergence values to the CRANDIO output file.

Input

All required input variables are supplied to DIVERG from ICEMDL through the formal parameter list. No new variables are input.

Processing

Subroutine DIVERG operates like all output module subroutines. Major processing functions are performed for the sole purpose of setting up the data and appropriate record names for the CRANDIC data base.

The data is placed into the common block array DIVRG. The record label is set up using a data statement which specifies the catalog name, tau value and record length.

If an error occurs on the CWRITER function an error message is written, stating this fact and the program will terminate.

Output

Subroutine DIVERG places the divergence values on the CRANDIO output file.

Interfaces

Subroutine DIVERG interfaces with the standard FNOC CRANDIO software.

DIVERG is called on every fourth time step by ICEMDL.

Tables and Items

The following common block is defined;

/DV/ - MDV - contains CRANDIO ID block;
DIVRG - contains data values;
FILLV - fills the small page (requirement of CRANDIO).

2.4.4.2 Subroutine UVPLOT

Subroutine UVPLOT is used to output the ice drift forecasts.

Input

All input to UVPLCT is provided by ICEMDL, through the parameter list. No new variables are introduced.

Processing

The first main function of UVPLOT is to convert the U, V ice drift components to ice drift direction and speed. This is accomplished through the FNOC routine UVDDFF. Ice drift speed is converted from m/sec to knots.

The final function is setting up of the CRANDIO record lables for CRANDIC. The record labels are set up using a data statement containing the catalog names, tau values and record lengths.

If an error occurs on the CWRITER function, a message is written and the program terminates.

Output

The ice drift direction and speed is output to the final CRANDIO file.

Interfaces

UVPLCT interfaces with the FNCC CRANDIC software. UVPLCT is called on every fourth time step by ICEMDL.

Tables and Items

No new major variables are introduced by UVPLOT.

2.4.4.3 Subroutine BUCYD

Subroutine BUOYD is used to calculate the drift of simulated buoys placed in the model.

Input

All input required for BUOYD is supplied by ICEMDL through the parameter list. No new variables are introduced by BUOYD.

The buoy drift positions are input through common block /BUOY/.

Processing

The initial processing of BUCYD checks the position of each buoy. This is done to determine if a buoy has moved out of the grid. Once a buoy reaches a boundary it is no longer tracked. The routine branches around the remainder of the processing and goes on to another buoy when one has moved to the boundary.

Subroutine INTRP, a FNOC utility routine, is used to interpolate the U, V ice drift components to the buoy position recorded during the previous time step. The u, v values are then used to calculate how far the buoy shall have moved during the current time step. This distance is then determined in terms of grid units on both the model grid and the FNOC hemispheric grid.

The final position of each buoy, in terms of both grid, is printed in a table.

Output

The positions of the buoys, calculated in common block /BUOY/ are output from BUOYD a printed table, outlining the position of each buoy is provided.

Interface

BUOYD utilizes the FNOC utility routine INTRP to provide u, v components of ice drift at buoy positions.

Tables and Interfaces

No new variables are introduced within BUOYD.

2.4.4.4 Subroutine HAPLOT

Subroutine HAPLOT is used to output the ice thickness and concentration values to the CRANDIC file.

Input

All required input to HAPLOT is provided by ICEMDL through the formal parameter list. No new input variables are defined.

Processing

Processing within HAPLOT performs two functions. The first is a unit conversion while the second creates the necessary data for CRANDIC. Ice thickness is output first. The thickness values are converted from meters to cm. The CRANDIC record labels and data are set up into their respective positions and CWRITER is used to write the data.

The ice concentration values are handled in the same manner. However no unit conversion is performed.

If an error results in the processing of any CWRITER an appropriate message is written and the program shall terminate processing.

Output

The ice thickness and compactness values are output to the CRANDIO file.

Interfaces

HAPLOT interfaces with the CRANDIO software.

Tables and Items

No new major variables are introduced.

2.4.4.5 Subroutine GROWDEC

Subroutine GROWDEC is used to output the open water growth and ice growth forecasts.

Input

All required input is passed to GROWDEC by ICEMDL.

Processing

The processing of GRCWDEC proceeds exactly as other output module subroutines. The CRANDIO labels are defined and CWRITER is used to output the data.

Output

Forecasts of open water and total ice growth are written to the CRANDIO file.

Interfaces

GROWDEC interfaces with the CRANDIO software.

Tables and Items

Nom new variables are defined.

2.4.4.6 Subroutine PRNT

Subroutine PRNT is a small subroutine which is used to print model arrays. The processing of PRNT depends entirely upon the input data specifications.

Input

The formal parameters are defined as follows;

i) ARRAY - array to be printed;
ii) I,J,K - dimensions of ARRAY;
iii) MI,MZ - columns of ARRAY which are printed;
iv) N - number of rows of ARRAY to be printed.

Output

Printed output of a data array is provided.

Section 3.0 Environment

3.1 Equipment Environment

FNOC operates a multiprogramming/-multi-- mainframe computer system consisting of three CDC 6500's, with 131K each of central memory (CM), 9 CDC CYBER 170/175 with 196K CM and 1 million words of CDC 7030 extended core storage (ECS), related auxiliary equipment (7 and 9 track tape units, disk storage, etc.), a CYBER 203 with 2 million words of CM and its front-end processor CYBER 170/720, and a VARIAN plotter and its plotting software. The Sea Ice model is designed to run on the CYBER 203 computer, and its output is plotted on the VARIAN plotter.

3.2 Support Software

FNOC operates under the NOS/BE operating system. This system contains many local enhancements/modifications to facilitate ease of operation. Most of the enhancements are documented in either the FNOC subroutine/utility file or the FNWC Computer User Guide Edition 2. The Sea Ice Model was converted to CDC CYBER 200 FORTRAN Language, version 1.5 and utilizes various routines available on FNWCLIB.

3.3 Data Base

A number of data bases are maintained and needed by the sea ice model.

3.3.1 General Characteristics

Three separate input data bases are required for the sea ice model to properly execute. Two of these are created by the user. The third data base consists of the FNOC environmental data. Currently this data is not operationally available on the CY203, therefore, this data base is also set up by the user, from data available on other machines.

i) TAPE7

TAPE7 is the mnemonic for the main input file. This file is set up by a user and remains permanent for the desired configuration of the model runs.

The file is utilized by the input module and describes the execution configuration for the current run (e.g., grid size, grid location, grid configuration).

ii) TAPE8

TAPE8 is a more dynamic data file than TAPE7. TAPE8 contains the DTG's of the days during the current run. Therefore, this file will change for each run.

The file, TAPE8, needs to be defined by the user of the sea ice model.

iii) FNOC Data

FNOC data is kept on a CRANDIO file which, at the time of this writing, must be set up by the user. These data will change for a specific run. The data is obtained from another machine which has access to the FNOC master data base (MASFNWC).

iv) Output Data-Base

A CRANDIC output data base is maintained by the sea ice model. This data base is filled with forecast variables computed by the sea ice model. Variables are written in specified time steps, defined in routine ICEMDL.

3.3.2 Organization and Detailed Description

i) TAPE7

The file labled TAPE7 is a binary file accessed by a formatted READ. The following information is contained, with formats;

i) grid size specifications - 2I5;
ii) FNOC i grid points for MESH - 2F10.2;
iii) FNOC j grid points for MESH - 2F10.2;
iv) number of rows/columns in model grid - I5
v) uv boundary mask - 27F2.0, 27 rows
vi) thickness boundary mask - 28F2.0, 28 rows;
vii) outflow boundary mask - 28F2.0, 28 rows;
viii) ocean current direction - 25F3.0, 25 rows.

ii) TAPE8

The file labeled, TAPE8, is a binary

file accessed with a formatted READ. The following information is contained, with formats;

i) number of time steps to be run - I5;

ii) DTG, one row for each time step 3(A8,2X).

The DTG rows are defined as described under ICEMDL.

iii) FNOC Input Data

The input FNOC data base is a CRANDIC data file on the CY203. CRANDIO is the operational data file format, specified by FNOC, on the CY203. Detailed characteristics of the CRANDIO files can be found in the CRANDIO software documentation distributed by FNOC.

The data on the CRANDIC file is created by transferring ZRANDIC data, from other FNCC machines to the CY203 with proper ZRANDIO to CRANDIO conversion software.

The following records are contained on the CRANDIO input file;

Record	Contents
A20	u-wind component
A21	v wind component
A10	air temperature
A01	surface pressure
A12	surface vapor pressure
A11	short wave radiation incoming
A18	total heat flux
A16	sensible plus evaporative heat flux.

iv) CRANDIO Output File

The CRANDIO output file is created by the sea ice model. CRANDIC records are written consisting of forecast variables on certain time steps. The following records are written every four time steps;

Record		Contents
i)	DIV	ice divergence/convergence
ii)	FFF	ice drift speed
iii)	DDD	ice drift direction
iv)	THK	ice thickness
v)	CCM	ice concentration
vi)	PRS	ice pressure
vii)	GAR	open water growth
viii)	HDF	ice growth.

Section 4.0 Program Maintenance Procedures

4.1 Conventions

The Sea Ice Model system adheres to structured design and programming principles. Flowcharting and naming conventions adhere to the standard identified below.

a. FNWC User Guide, Edition 2, February, 1974.

b. CDC Programming Standards, CDC-STD 1.80.000, December, 1971.

4.2 Verification Procedures

The methods of verifying the sea ice mode: output through display of the output file on plotting display o. printed output. Plotting of ice drift, ice growth/decay rate, ice thickness is a very efficient method used to check output of the model.

4.3 Error Conditions

This section describes the error conditions determined by the Sea Ice Model.

ICEMDL

Message - "OPEN ERROR"
Reason - error in opening the CRANDIC output file

RELAX

Message - "No convergence after 1300 iterations"

Reason - UICE and VICE are not convergent after 1300 iterations.

UVPLOT

Message - "STATUS is (value) ON WRITE OF filename"

Reason - ISTAT is not equal to 0.

Result - Output of UVPLOT is not written to CRANDIC file, program stops.

HAPLOT

Message - "STATUS IS (value) ON WRITE OF filename"

Reason - ISTAT is not going to 0.

Result - Output of HAPLOT is not written to CRANDIO file, program stops.

DIVERG

Message - "CWRITE STATUS IS (Value) ON WRITE OF filename"

Reason - ISTAT is not equal to 0, the CRANDIO output file is not opened.

Result - Output of DIVERG is not written to CRANDIO file, program stops.

INITIAL

Message - "CHECKNC no data initial"
Reason - No required data field for input
Result - Stop the program.

4.4 Special Maintenance Procedures

There are no special maintenance procedures for the Sea Ice Model program.

4.5 Special Maintennance Programs

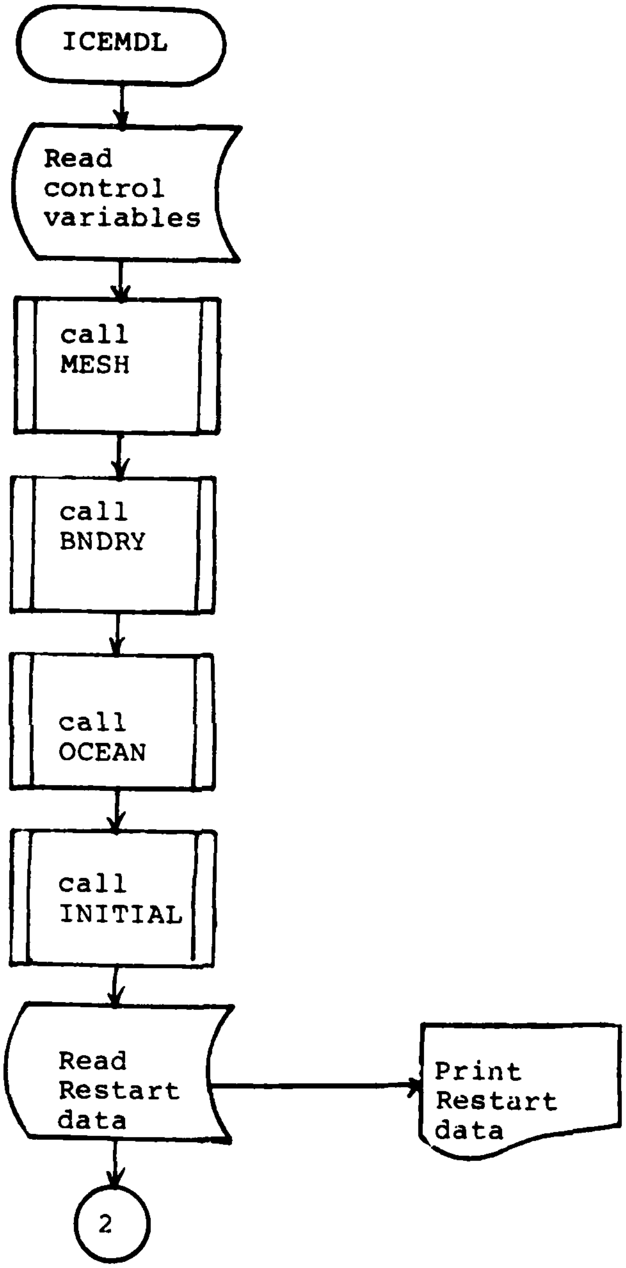
There are no special maintenance programs for the Sea Ice Model.

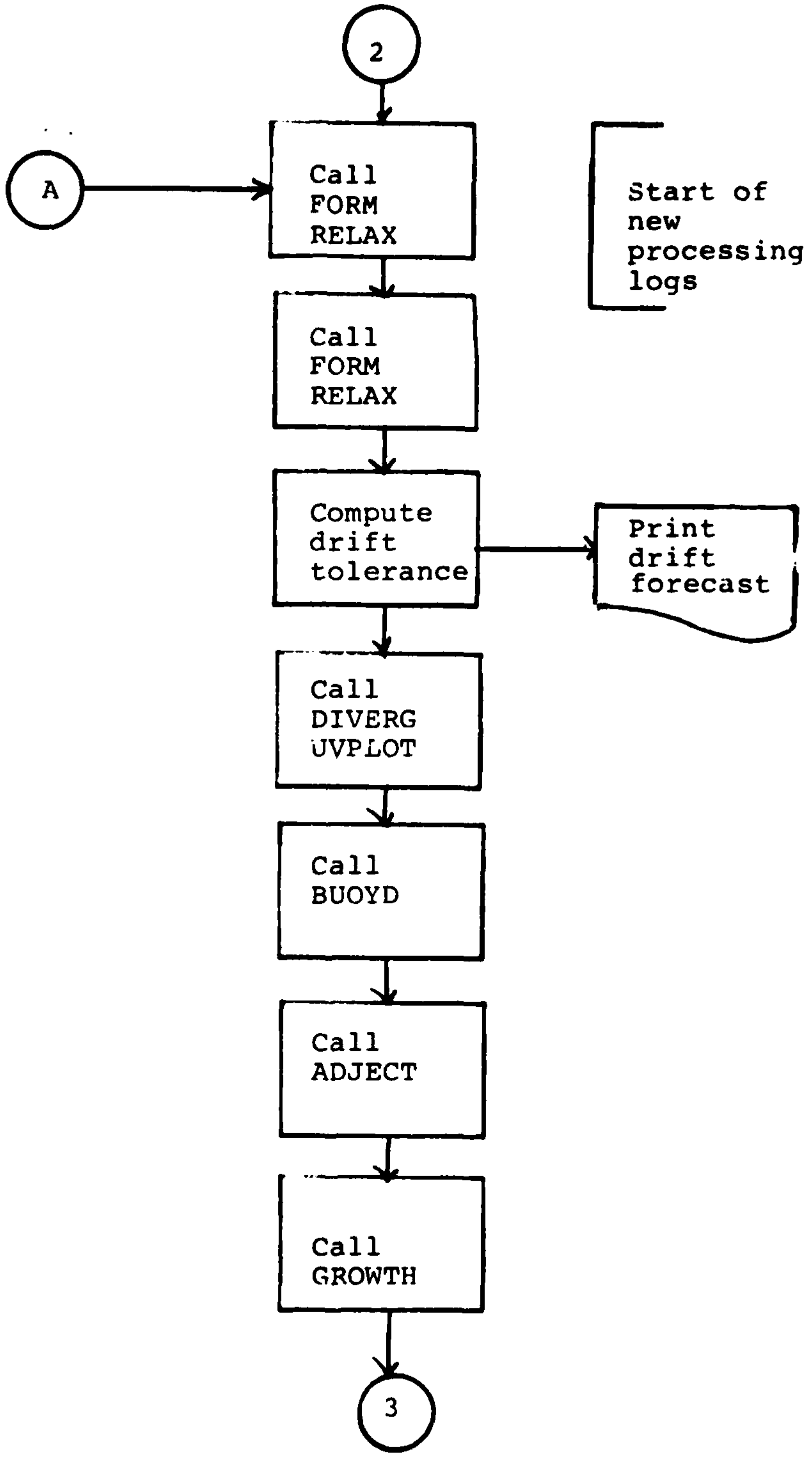
4.6 Listings

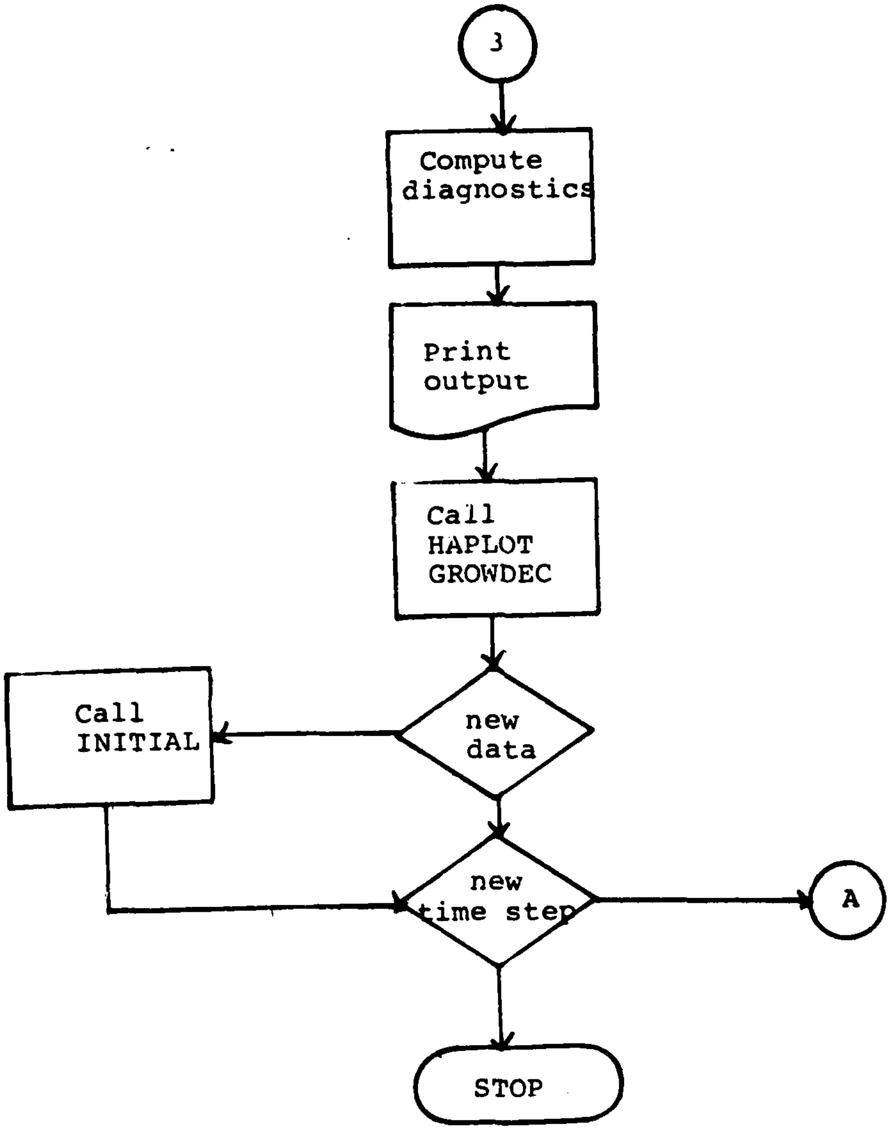
Listings of the Sea Ice Model program and subroutines are to be found in Appendix C.

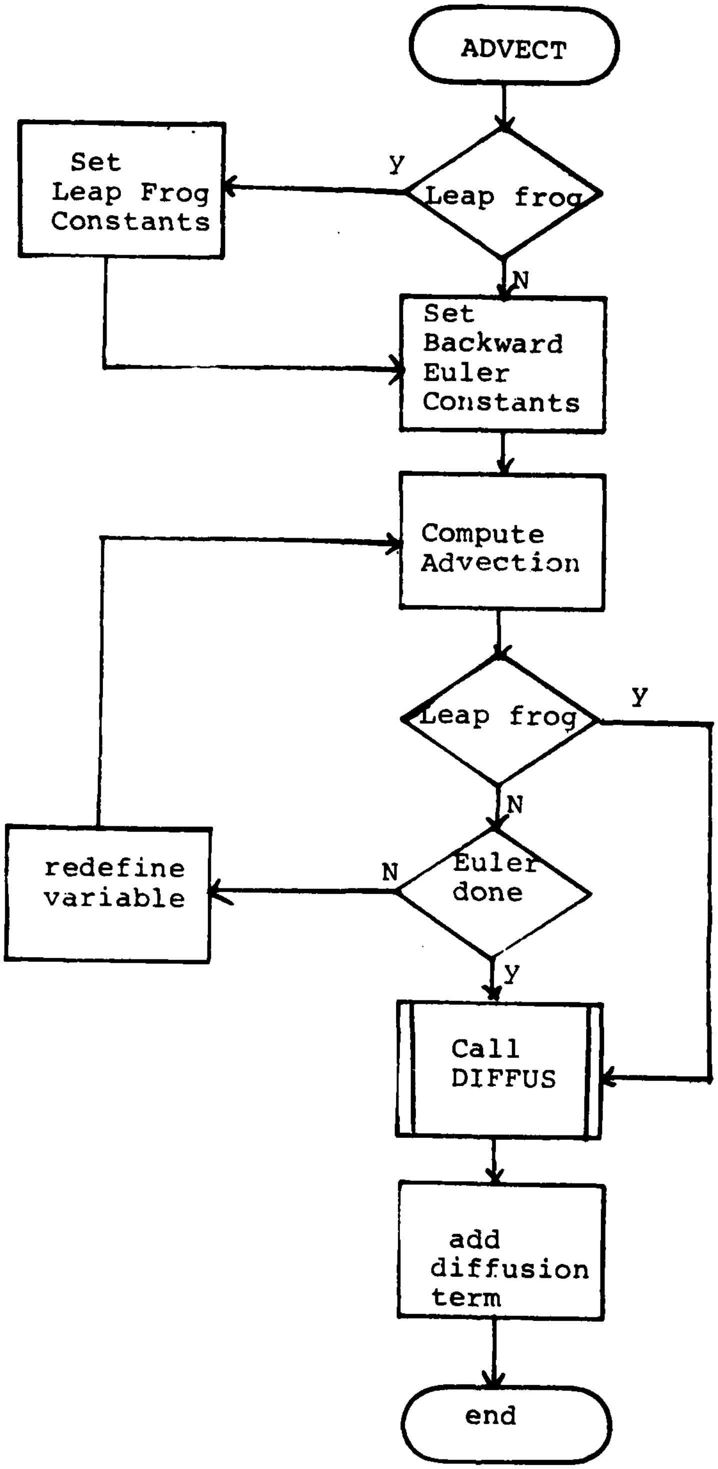
APPENDIX A

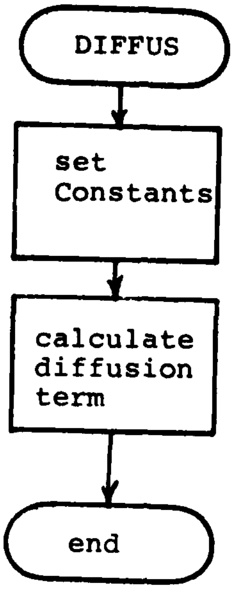
ICEMDL Flowcharts

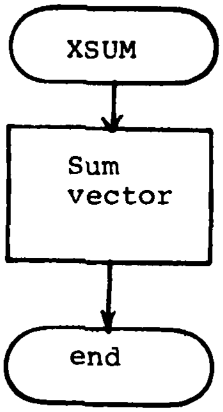

ICEMDL
Read control variables
call MESH
call BNDRY
call OCEAN
call INITIAL
Read Restart data
Print Restart data
2

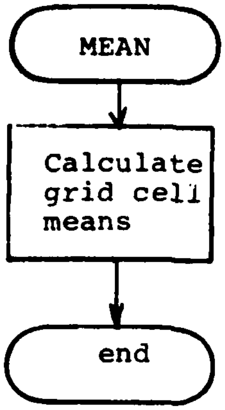

2
A
Call FORM RELAX
Start of new processing logs
Call FORM RELAX
Compute drift tolerance
Print drift forecast
Call DIVERG UVPLOT
Call BUOYD
Call ADJECT
Call GROWTH
3

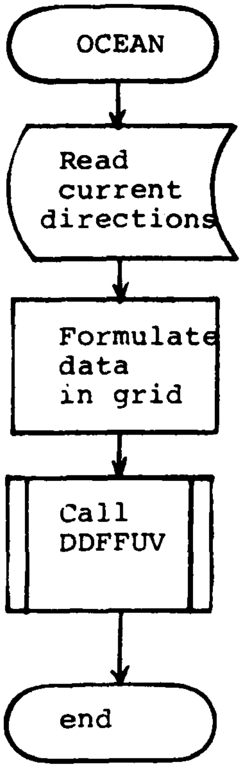

3
Compute diagnostics
Print output
Call HAPLOT GROWDEC
Call INITIAL
new data
new time step
A
STOP

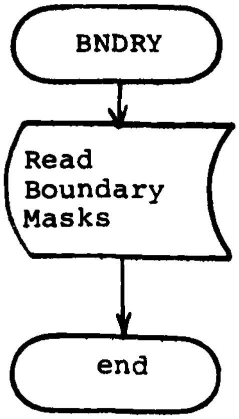

ADVECT
Leap frog
y
Set Leap Frog Constants
N
Set Backward Euler Constants
Compute Advection
Leap frog
y
N
Euler done
N
redefine variable
y
Call DIFFUS
add diffusion term
end

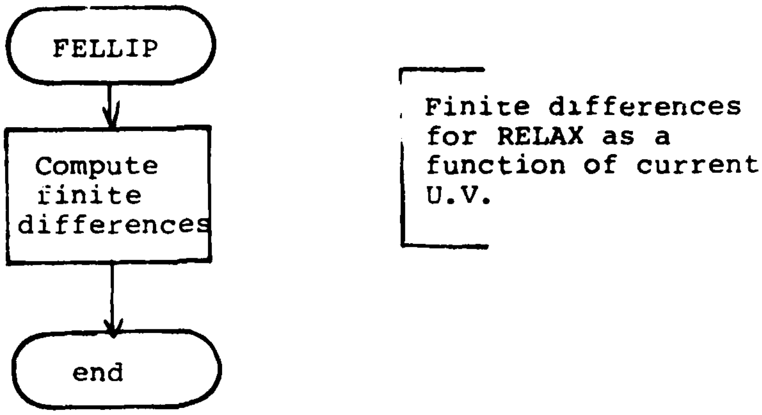

DIFFUS
set Constants
calculate diffusion term
end

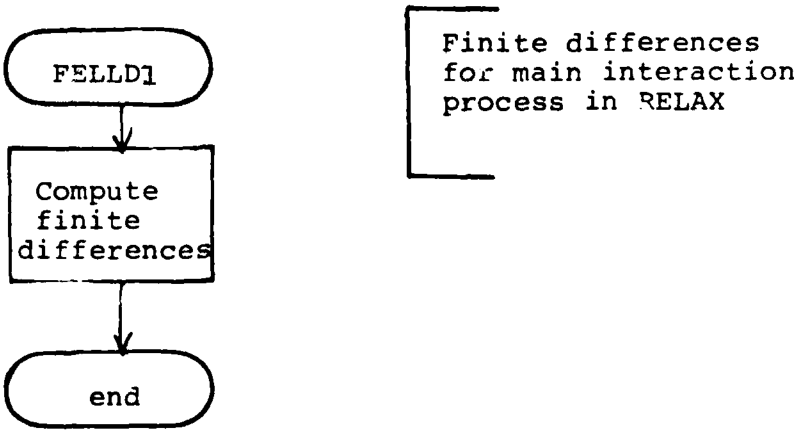

XSUM
Sum vector
end

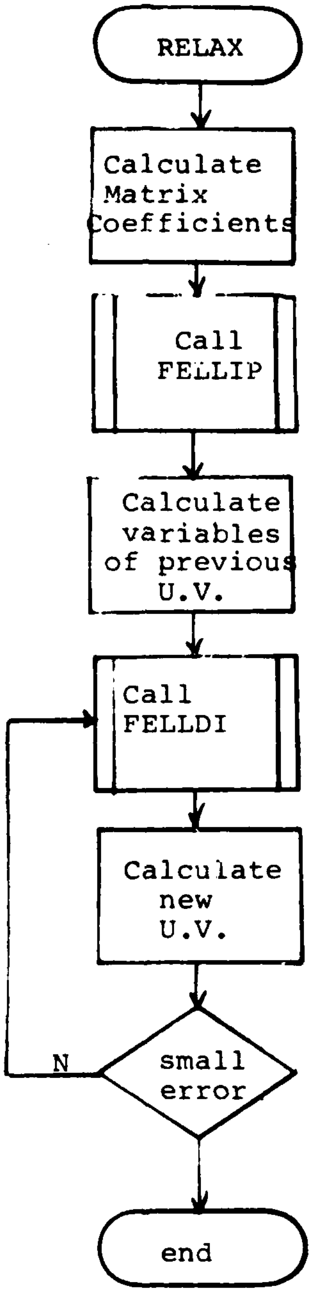

MEAN
Calculate grid cell means
end

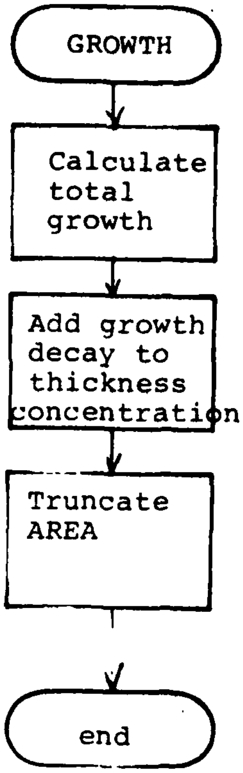

OCEAN
Read current directions
Formulate data in grid
Call DDFFUV
end

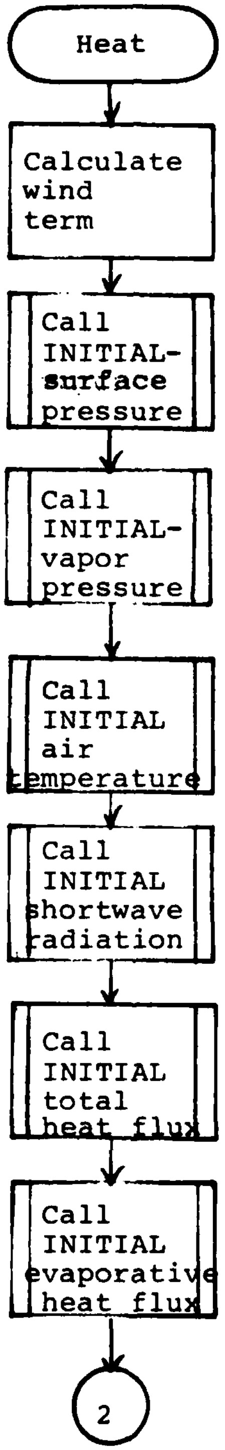

BNDRY
Read Boundary Masks
end

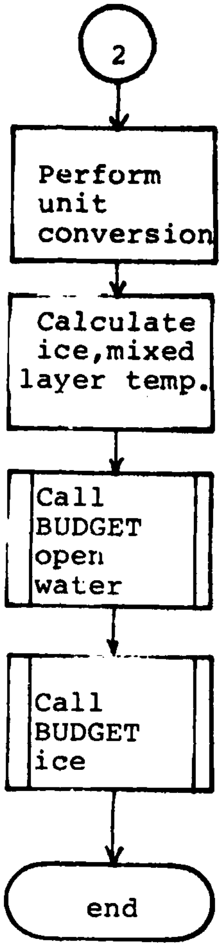

FELLIP
Compute finite differences
end
Finite differences for RELAX as a function of current U.V.

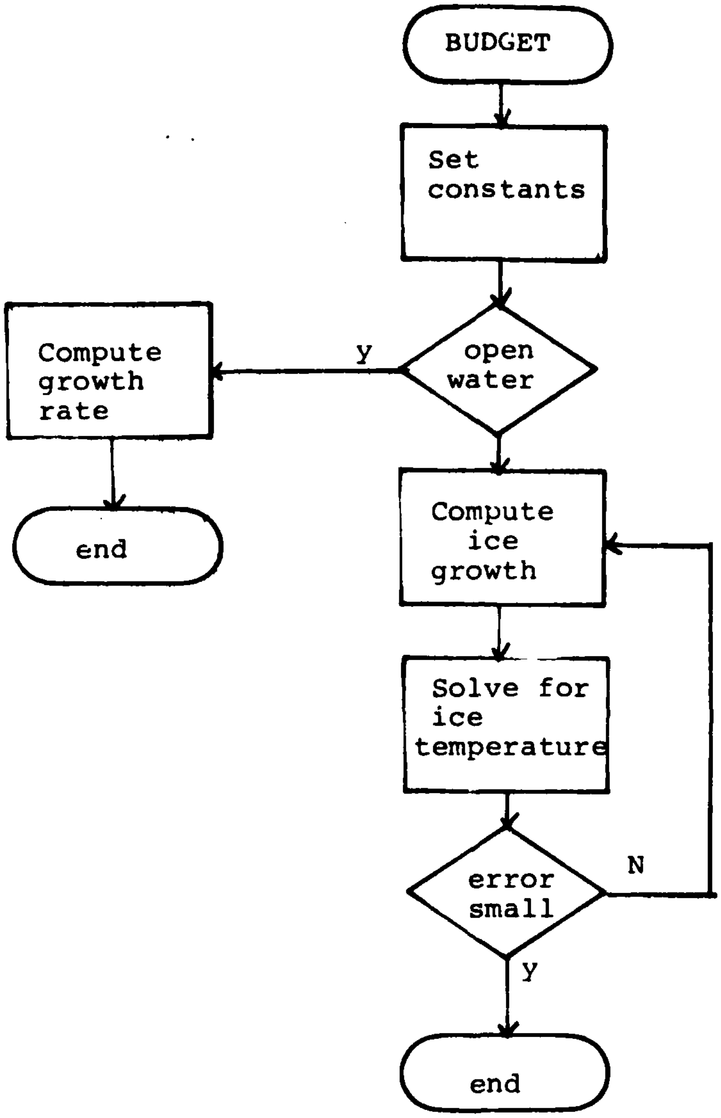

FELLD1
Compute finite differences
end
Finite differences for main interaction process in RELAX

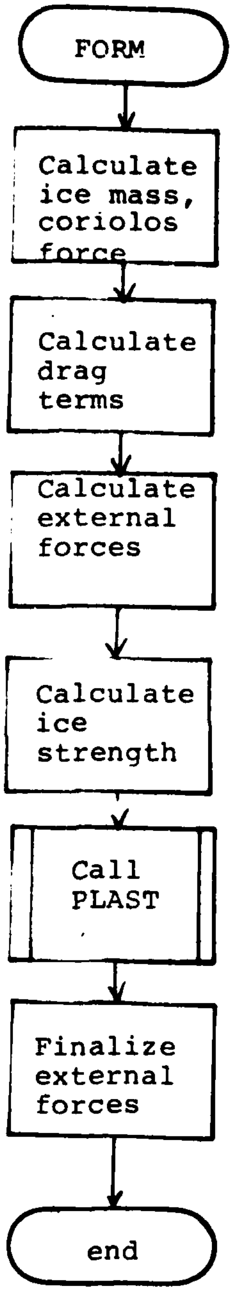

RELAX
Calculate Matrix Coefficients
Call FELLIP
Calculate variables of previous U.V.
Call FELLDI
Calculate new U.V.
N
small error
end

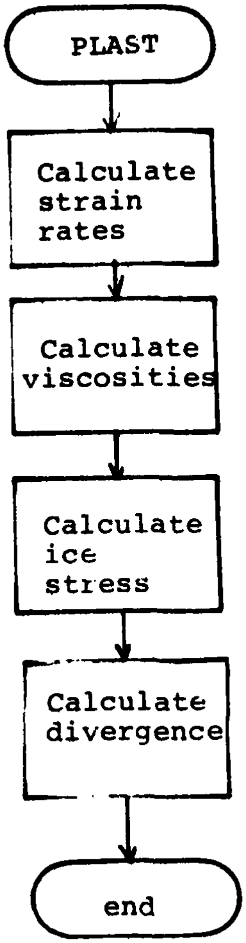

GROWTH
Calculate total growth
Add growth decay to thickness concentration
Truncate AREA
end

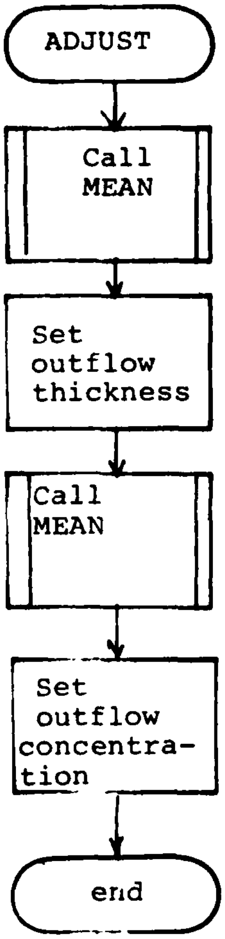

Heat
Calculate wind term
Call INITIAL-surface pressure
Call INITIAL-vapor pressure
Call INITIAL air temperature
Call INITIAL shortwave radiation
Call INITIAL total heat flux
Call INITIAL evaporative heat flux
2

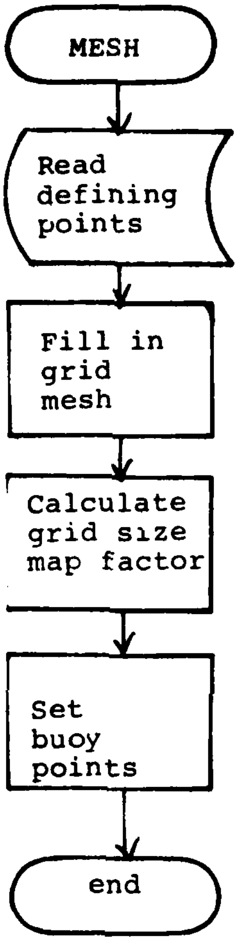

2
Perform unit conversion
Calculate ice,mixed layer temp.
Call BUDGET open water
Call BUDGET ice
end

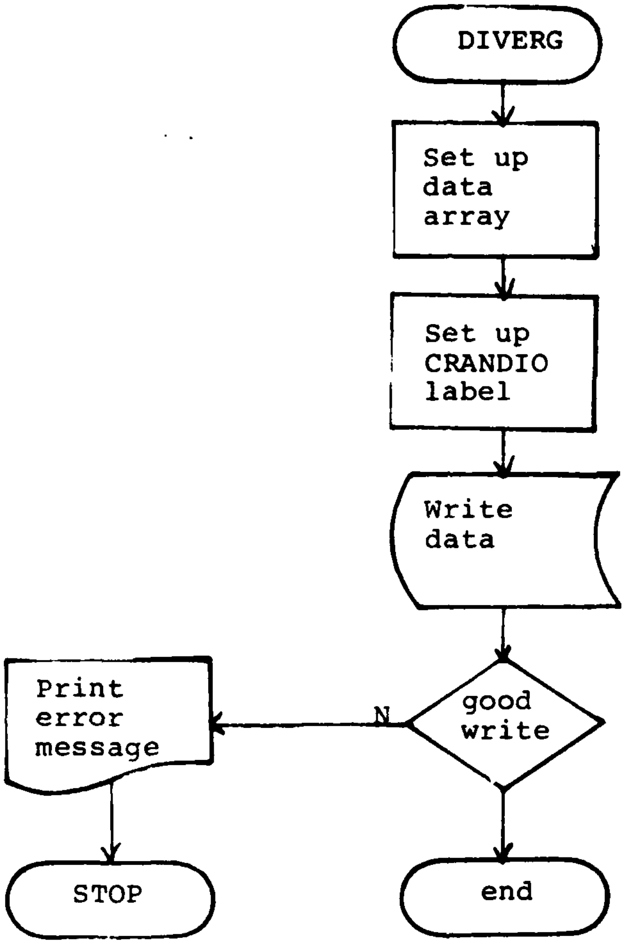

BUDGET
Set constants
open water
y
Compute growth rate
end
Compute ice growth
Solve for ice temperature
error small
N
y
end

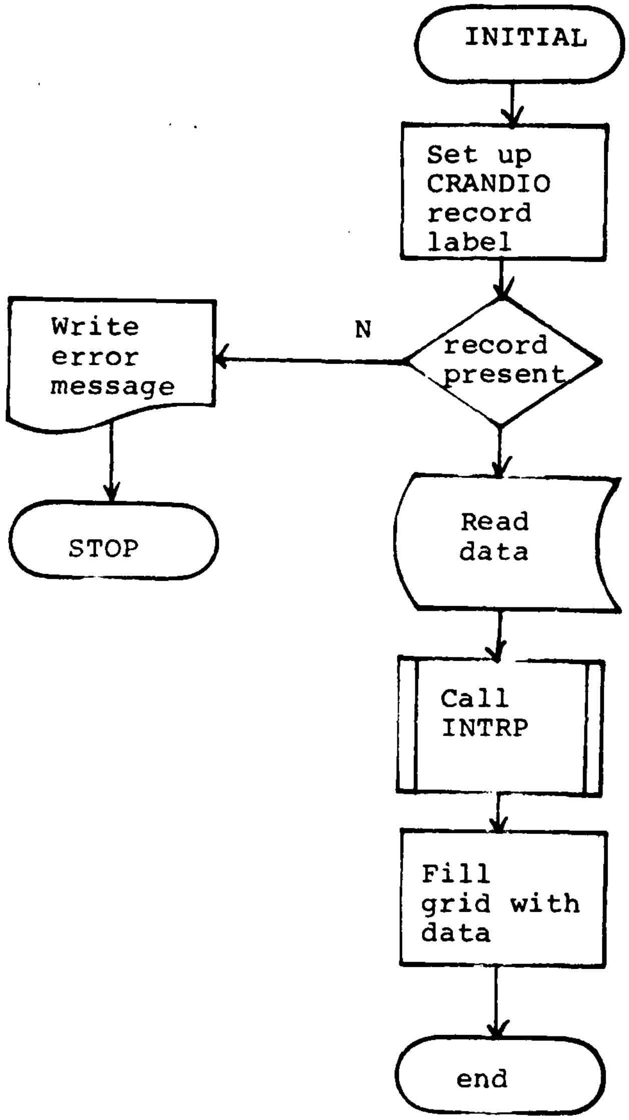

FORM
Calculate ice mass, coriolos force
Calculate drag terms
Calculate external forces
Calculate ice strength
Call PLAST
Finalize external forces
end

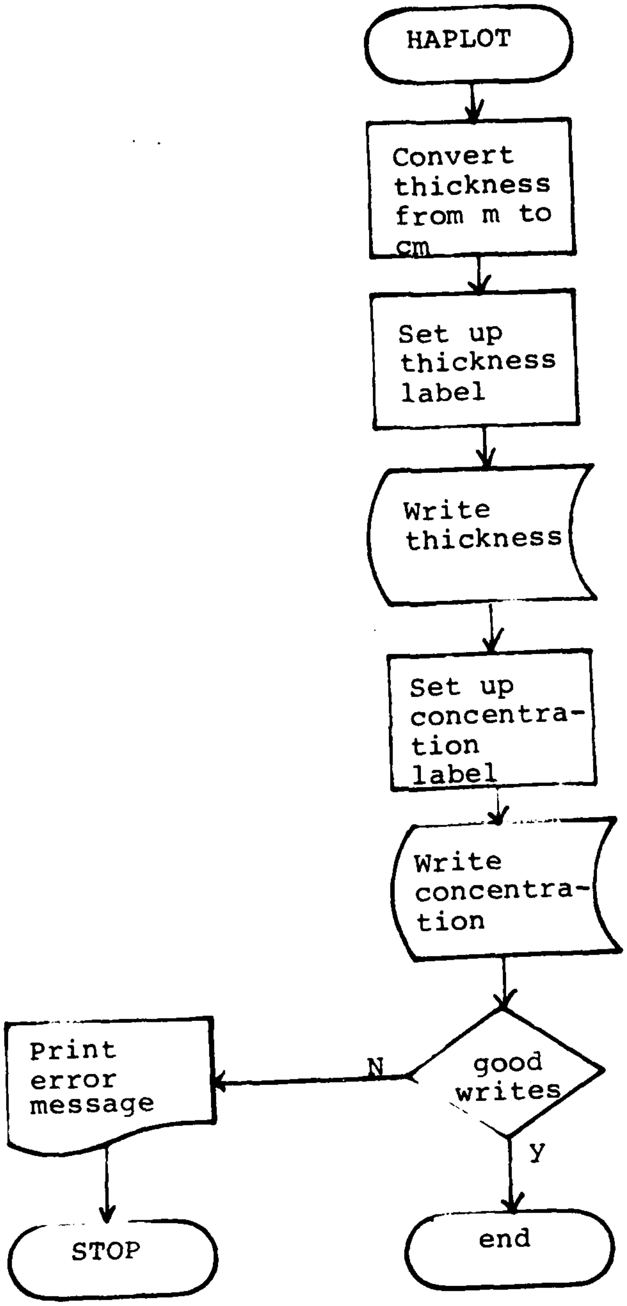

PLAST
Calculate strain rates
Calculate viscosities
Calculate ice stress
Calculate divergence
end

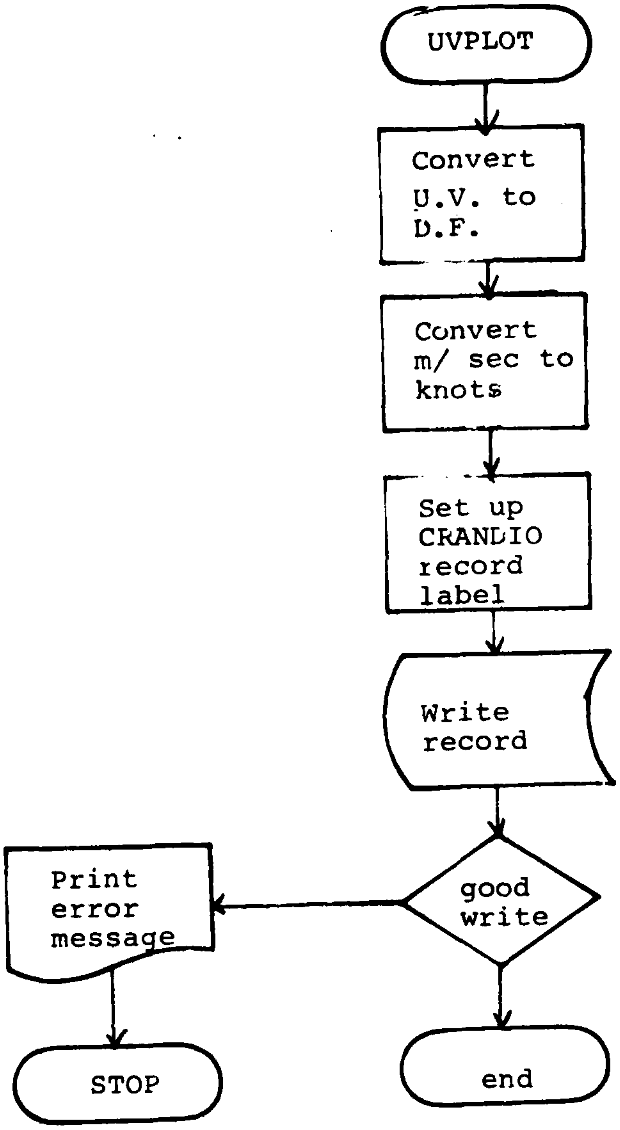

ADJUST
Call MEAN
Set outflow thickness
Call MEAN
Set outflow concentration
end

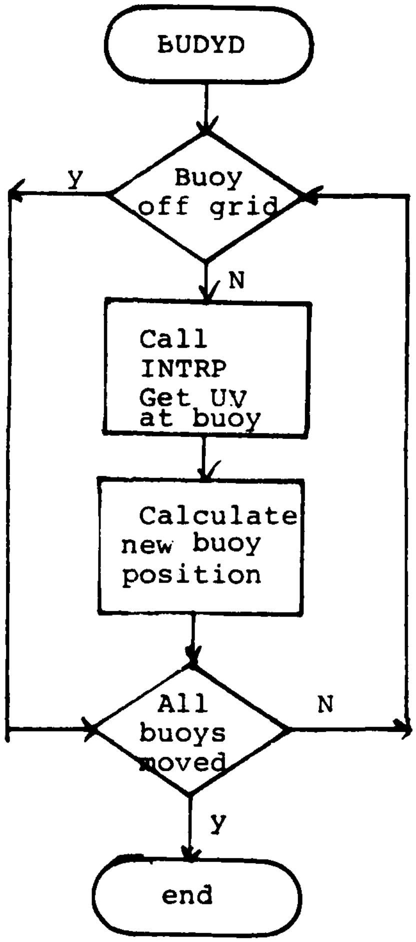

MESH
Read defining points
Fill in grid mesh
Calculate grid sıze map factor
Set buoy points
end


DIVERG
Set up data array
Set up CRANDIO label
Write data
good write
N
Print error message
STOP
end


INITIAL
Set up CRANDIO record label
record present
N
Write error message
STOP
Read data
Call INTRP
Fill grid with data
end


HAPLOT
Convert thickness from m to cm
Set up thickness label
Write thickness
Set up concentration label
Write concentration
good writes
N
Y
Print error message
STOP
end


UVPLOT
Convert U.V. to D.F.
Convert m/ sec to knots
Set up CRANDIO record label
Write record
good write
Print error message
STOP
end

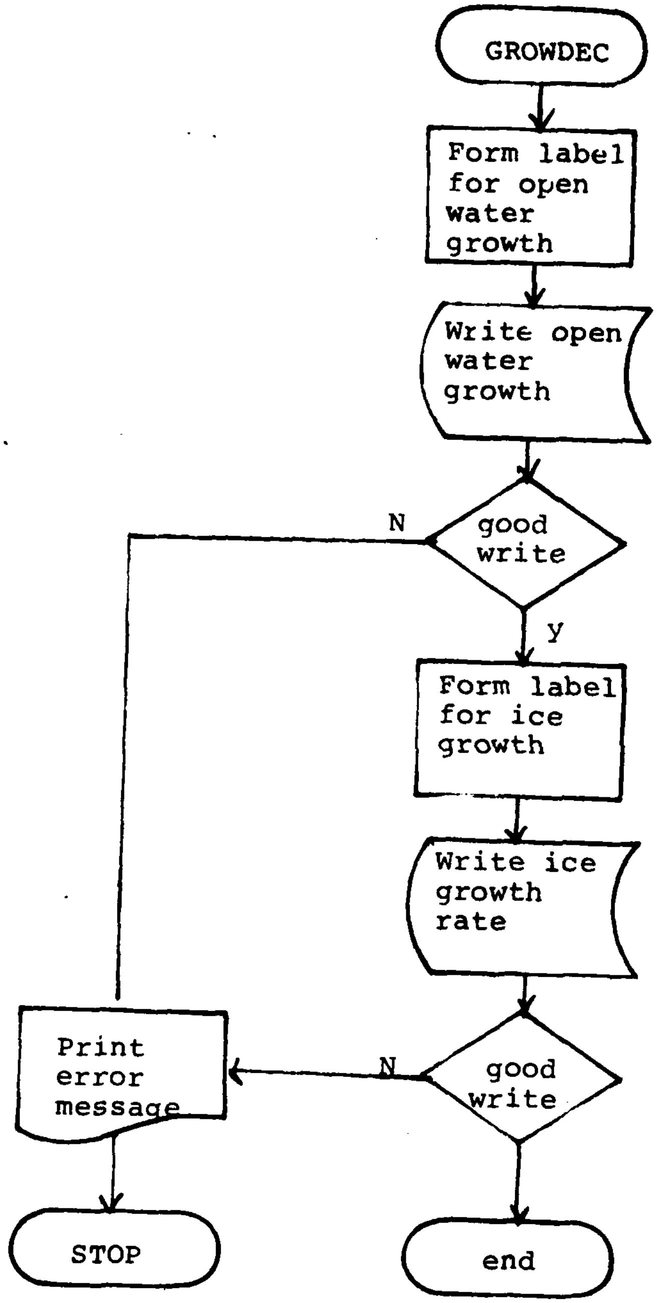

BUOYD
Buoy off grid
y
N
Call INTRP Get U,V at buoy
Calculate new buoy position
All buoys moved
N
y
end

AD-A114 343   CONTROL DATA CORP   MONTEREY CA   F/G 8/12
PROGRAM MAINTENANCE MANUAL POLAR ICE FORECAST SUBSYSTEM - ARCTI--ETC(U)
OCT 81   P A HARR, T C PHAM, J P WELSH   N00014-81-F-0028

UNCLASSIFIED   NORDA-TN-122   NL

2 OF 2
AD
A114343

END
DATE
FILMED
5-82
DTIC


GROWDEC
Form label for open water growth
Write open water growth
good write
N
y
Form label for ice growth
Write ice growth rate
good write
N
Print error message
STOP
end

Appendix B

Grid Structure

The model grid is set up in a staggered manner. Momentum variables are defined at the grid points while thermodynamic variables are defined for grid cells.

Environmental variables accessed from the FNOC data base are computed to be valid at the grid cell locations. Subventive AVG is used to compute the grid cell averages for the thermodynamic variables.

The following example illustrates the definition of grid parameters for a small grid.

Say, we define NX,NY to be 7 (dimension of momentum variable grid). Figure B1 illustrates that NX1,NY1 (thermodynamic grid) will be 8 while the value of NUMBER will be 9.

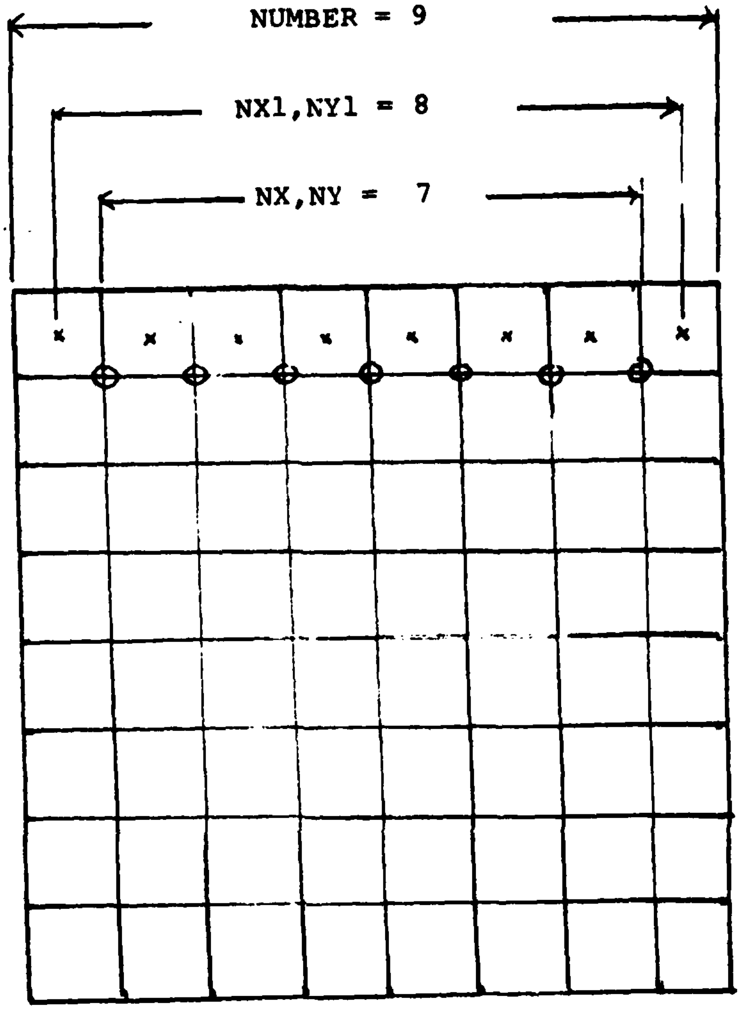


Figure B1   Grid Configuration

APPENDIX C

ICEMDL Listings

```
FTN 5.1 CYCLE FTN1835  BUILT 08/03/81 23 27    SOURCE LISTING
            PROGRAM ICEMDL(INPUT,OUTPUT,TAPE8=SUMS,TAPE7=IN,TAPE6=OUTE)
      C****************************************************************
      C
      C  MAIN DRIVING PROGRAM FOR VISCOUS PLASTIC SEA ICE MODEL
      C
      C****************************************************************
      C
      C     PRIMARY PROGNOSTIC VARIABLES ARE DEFINED IN THE FOLLOWING
      C     DIMENSION STATEMENT. COMMON BLOCKS ARE USED TO HOLD VARIOUS
      C     DIAGNOSTIC VARIABLES AND WORKING STORAGE ARRAYS.  THESE
      C     ARRAYS ARE EQUIVALENCED TO PRIMARY VARIABLES AND OTHER WORKING
      C     ARRAYS.
      C
            DIMENSION UICE(27,27,3), VICE(27,27,3), ETA(24,24),
           *          ZETA(24,24), SLIDX(24,24), SLIDY(22,23),
           *          OUT2(24,24),AMASS(27,27),
           *          HEFF(24,24,3), AREA(24,24,3),
           *          UICEC(27,27), VICEC(27,27),GWATX(27,27),
           *           GWATY(27,27),STRESS(24,24,4)
           *.         FHEFF(24,24),FO(24,24),HEFFM(24,24),UVM(27,27),
           *          OUT(24,24),OAREA(24,24),DHEFF(24,24)
           *.        SWIT(24,24), SWOD(23,24)
           *.         TOTS(3), DIV(25,25)
            DIMENSION HOTEF(24,24), HCOOL(24,24), SAVEA(24,24)
      C
      C
      C
            COMMON /WINDY/ WINDXI(20), WINDYI(20), RX(20), RY(20)
            COMMON /TIME/ DELTS,FDAYS,COUNTS,MONTHS,HEATS,MESHS,UNITS
            COMMON  /FORCE/  FORCEX(27,27), FORCEY(27,27)
            COMMON /STEP/ DELTAT, DELTAX, DELTAY, DELTA1, DELTA
            COMMON /PRESS/ PRESS(24,24)
            COMMON /BUFFER/ IDENT(26),DATA(63,63),FILL(107)
            COMMON /TU/ IJHR(414)
            COMMON /DD/ ID(24), DD(624), FILLD(375)
            COMMON /EE/ IE(24), EE(625), FILLE(375)
            COMMON /AR/ IDA(24), ARRAY(625), FL(375)
            CHARACTER*80 ID,IE,IDA,LABEL,TD,ID2,ID3,ID4
            DATA ID1 /8HTHK*  24/
           *.    ID2 /8HCON*  24/
           *.    ID3 /8HPOOL*  24/
           *.    ID4 /8HHEFF*  24/
           *.    ID(3) /8H     648/
           *.    IE(3) /8H     648/
           *.    IDA(3) /8H     648/
      C
      C
      C
            CHARACTER*80  IDENT,LABEL,IDTS
      C
      C
      C
            DATA
           *  (WINDXI(L),L=1,20) /30.4, 29.5, 31.7, 0.0, 32.7, 29.1,
           *                       0.0,  0.0, 31.0, 0.0, 29.7,  0.0,
           *                      27.0, 27.4,  0.0, 32.0,  0.0, 24.3,
           *                       0.0, 29.2/
           *  (WINDYI(L),L=1,20) /32.2, 31.1, 31.0,  0.0, 31.4, 24.5,
           *                       0.0,  0.0, 31.1,  0.0, 31.2,  0.1,
           *                      24.4, 26.5, 0.0, 31.4, 0.0, 23.3,
           *                              0.0, 32.4/
           DATA LAD,ICOUNT,IOUT,HESUM,UVSUM,HHESUM/0.0,0.0,4*0.0/
           DATA DELTAT,DELTT,ERROR,EHSUM1,GRSUM1,ARSUM1,ARR,HO/2*86400.,1.,
           *        0.000001,3*0.0, 0.15, 0.5/
           CALL SECOND(THEGIN)
           READ (8,13) ITSTEP
           READ (7,13) NX,NY,NX1,NY1,N3,N4
     13    FORMAT(6I5)
           PRINT 10000, NX,NY,NX1,NY1,N3,N4
   10000   FORMAT(1X,'INPUT GRID     ',6I5)
   C
   C NOW DECIDE ON BASIC PARAMETERS
   C
           READ (8,14) IDTG(1), IDTG(2), IDTG(3)
     14    FORMAT(A8,2X,A8,2X,A8)
           CALL MESH(GRDI,GRDJ,NX,NY,NX1,NY1,NUMBER)
           ERROR=5.0*ERROR
           DIFF1=.002*DELTAX
           DIFF1=2.0*DIFF1
   C
   C DOUBLE HO BECAUSE OF MOD IN GROWTH
   C
           HO=2.0*HO
   C
   C NOW DEFINE BOUNDARIES
   C
           CALL BNDRY(HEFFM,UVM,OUT,NX,NY,NX1,NY1)
           DO 122 J=1,NY1
           DO 122 I=1,NX1
           HESUM=HESUM+OUT(I,J)
           HHESUM=HHESUM+HEFFM(I,J)
    122    CONTINUE
           DO 125 J=1,NY
           DO 125 I=1,NX
           UVSUM=UVSUM+UVM(I,J)
    125    CONTINUE
           PRINT 6,HESUM,UVSUM
           PRINT 6,HHESUM,UVSUM
      6    FORMAT(1X,*HESUM IS*,G15.6,*UVSUM IS*,G15.7)
   C
   C       FORM OCEAN CURRENTS
   C
           CALL OCEAN(GWATX,GWATY,NX,NY,GRDI,GRDJ,UVM)
   C
   C       OPEN THE CRANDIO OUTPUT FILE
   C
           IFILE = 6HTAPE3
           CALL COPEN(IFILE,ISTAT)
           IF(ISTAT .NE. 0) STOP 'COPEN ERROR'
           ITAU = 0
           MSH = MESHS
           CALL SECOND(MESHS)
           MESHS = MESHS - MSH
   C
   C       OBTAIN THE INITIAL WIND VALUES
   C
           LSTEP = 0
           CALL INITIAL(1,GAIRX,GRDI,GRDJ,IDTG,NUMBER,ITAU)
           CALL INITIAL(2,GAIRY,GRDI,GRDJ,IDTG,NUMBER,ITAU)
   C
```

```
      C      CONVERT WINDS FROM CM/SEC TO M/SEC
             N2 = NX + 2
             DO 10 I = 1,N2
             DO 10 J = 1,N2
             GAIRX(I,J) = GAIRX(I,J) / 100.0
             GAIRY(I,J) = GAIRY(I,J) / 100.0
          10 CONTINUE
      C
      C
      C NOW INITIALIZE SYSTEM
      C FIRST GUESS AT INITIAL HEFF AND AREA
      C
      C
             DO 12 J = 1,NY
             DO 12 I = 1,NX
             UICEC(I,J) = 0.0
             VICEC(I,J) = 0.0
          12 CONTINUE
             DO 99 K = 1,3
             DO 99 J = 1,27
             DO 99 I = 1,27
             UICE(I,J,K) = 0.0
             VICE(I,J,K) = 0.0
          99 CONTINUE
             DO 101 J = 1,22
             DO 101 I = 1,24
             HEFF(I,J,3)=0.0
             HEFF(I,J,2)=0.0
             AREA(I,J,2)=1.0
             AREA(I,J,3)=1.0
             HEFF(I,J,1)=(3.0)/0.91
             HEFF(I,J,1)=HEFF(I,J,1)*OUT(I,J)
             AREA(I,J,1)=1.0
         101 CONTINUE
      C
      C      CALCULATE TOTAL BASIN ICE THICKNESS
      C         EXCEPT AT OUTFLOW CELLS
      C
             CALL XSUM(HEFF,THEFF,NX1,NY1)
             THETA=1.0
      C
      C      START WITH AN  INITIAL VELOCITY FIELD OF ZERO
      C
             CALL FORM(UICE,VICE,ETA,ZETA,AMASS,GAIRX,GAIRY,GWATX,GWATY,
            *     DRAGS,DRAGA,OUT,HEFFM,NX,NY,NX1,NY1,DIV ,HEFF,AREA)
      C
      C      SET MASS TO 0 AND DEFINE THE VISCOSITIES
      C
             DO 103 J = 1,27
             DO 103 I = 1,27
             AMASS(I,J)=0.0
             ZETA(I,J) = HEFF(I,J,1) * (1.0E+11)
             ETA(I,J)=ZETA(I,J)/4.0
         103 CONTINUE
      C
      C
             CALL RELAX(UICE,VICE,ETA,ZETA,DRAGS,DRAGA,AMASS,UVM
            *,ERROR,THETA,UICEC,VICEC,HEFFM,NX,NY,NX1,NY1)
             PRINT 551
         551 FORMAT(1H0,*UICE AND VICE AFTER FIRST RELAX *)
      C
      C
      C        SUBROUTINE ADJUST-
      C
      C
      C        THICKNESS AND COMPACTNESS VALUES AT THE OUTFLOW CELLS ARE
      C        ESTIMATED USING SUBROUTINE MEAN.
      C        ALL ICE FLOWING INTO THE GRID THROUGH THE OPEN CELLS IS
      C        ACCOUNTED FOR,
      C
      C        THEFF1  CONTAINS THE AMOUNT OF ICE IN THE OUFLOW CELLS....
      C        THIS IS USED IN THE ADVECTION CALCULATIONS.
      C
      C
               CALL ADJUST(HEFF,AREA,OUT,HEFFM,NX,NY,NX1,NY1)
      C
      C
      C
      C        NOW START THE STANDARD PREDICTOR CORRECTOR ITERATION SCHEME
      C
               CALL SECOND(TNEW)
               TT = TNEW - TBEGIN
               PRINT 9122,TT
         9122  FORMAT(1H0,********  INITIALIZATION TIME    *,E10.4)
          100  CONTINUE
               CALL SECOND(TNEW)
               CALL XSUM(HEFF,THEFF1,NX1,NY1)
               THEFF1 = THEFF1 - THEFF
      C
      C
      C        FIRST DO THE PREDICTOR
      C
               DO 121 J=1,NY
               DO 121 I=1,NX
               UICE(I,J,3)=UICE(I,J,1)
               VICE(I,J,3)=VICE(I,J,1)
               UICEC(I,J)=UICE(I,J,1)
               VICEC(I,J)=VICE(I,J,1)
          121  CONTINUE
               THETA=1.0
               DELTAT=DELTT/2.0
         1000  CONTINUE
               CALL FORM(UICE,VICE,ETA,ZETA,AMASS,GAIRX,GAIRY,GWATX,GWATY,
              *     DRAGS,DRAGA,OUT,HEFFM,NX,NY,NX1,NY1,DIV,HEFF,AREA)
               CALL RELAX(UICE,VICE,ETA,ZETA,DRAGS,DRAGA,AMASS,JVM
              *,ERROR,THETA,UICEC,VICEC,HEFFM,NX,NY,NX1,NY1)
      C
      C NOW DO REGULAR TIME STEP
      C
         1001  CONTINUE
               THETA=1.0
               DELTAT=DELTT
               CALL FORM(UICE,VICE,ETA,ZETA,AMASS,GAIRX,GAIRY,GWATX,GWATY,
              *     DRAGS,DRAGA,OUT,HEFFM,NX,NY,NX1,NY1,DIV,HEFF,AREA)
      C
      C NOW SET U(1)=U(2)=AND SAME FOR V
      C
               DO 111 J = 1,27
               DO 111 I = 1,27
               UICE(I,J,3)=UICE(I,J,1)
      VICE(I,J,3)=VICE(I,J,1)
      UICEC(I,J)=UICE(I,J,1)
      VICEC(I,J)=VICE(I,J,1)
      UICE(I,J,1)=UICE(I,J,2)
      VICE(I,J,1)=VICE(I,J,2)
  111 CONTINUE
      CALL RELAX(UICE,VICE,ETA,ZETA,DRAGS,DRAGA,AMASS,UVM
     *,ERROR,THETA,UICEC,VICEC,HEFFM,NX,NY,NX1,NY1)
C
C
      ICOUNT=ICOUNT+1
C
      SUI = 0.0
      SU = 0.0
      SMU = 0.0
      SMV = 0.0
      DO 130 J = 1,NY
      DO 130 I = 1,NX
C
C     SAVE THE T+1 TIME VALUES OF U AND V FOR USE IN ADVECTION OF
C     THE THICKNESS AND COMPACTNESS.
C
      UICEC(I,J) = UICE(I,J,1)
      VICEC(I,J) = VICE(I,J,1)
C
C     CALCULATE THE SQUARED VELOCITY AND THE SQUARED VELOCITY DIFFERENC
C     BETWEEN TIMES T + 1 AND T
C
      SU = SU + UICE(I,J,1)**2 + VICE(I,J,1)**2
      UERR = UICE(I,J,1) - UICE(I,J,2)
      VERR = VICE(I,J,1) - VICE(I,J,2)
      SUI = SUI + (UERR * UERR) + (VERR * VERR)
      SMU = AMAX1(ABS(UERR),SMU)
      SMV = AMAX1(ABS(VERR),SMV)
  130 CONTINUE
      SA = AMAX1(SMU,SMV)
C
C     PRINT THE VELOCITY INFORMATION
C
      PRINT 11,ICOUNT,IDTS(1)
   11 FORMAT(1H1,'THE TIME STEP IS ',I5,' THE DATE IS ',A8)
    9 FORMAT(*0*,(* *,5E20.14))
      PRINT *
      PRINT 8,SU,SUI,SA
    8 FORMAT(* *,*SQUARE VELOCITY, SU, VELOCITY DIFFERENCE,
     * MAX CHANGE *,/* *,3E20.12)
C
      IF(LSTEP .NE. 3) GO TO 647
      CALL DIVERG(DIV,NX,NY,GRDI,GRDJ,IDTS,ITAU)
      CALL UVPLOT(UICEC,VICEC,IDTS,NX,NY,GRDI,GRDJ,ITAU)
  912 CONTINUE
      CALL BNDYD(UICEC,VICEC,GRDI,GRDJ,NX,NY)
C
C NOW DO ADVECTION
C
C     CALCULATE THE ADVECTION OF THE THICKNESS AND COMPACTNESS
C
      CALL ADVECT(UICEC,VICEC,HEFF,DIFF1,LAD,HEFFM,NX,NY,NX1,NY1)
 1145 CALL ADVECT(UICEC,VICEC,AREA,DIFF1,LAD,HEFFM,NX,NY,NX1,NY1)
C
```

```
C     NOW DO THE GROWTH COMPONENT
C
C
C     SUBROUTINE HEAT CALCULATES THE GROWTH RATES FOR ICE AND OPEN
C     WATER.... USING A HEAT BUDGET.
C
      CALL HEAT(GRDI,GRDJ,HEFF,AREA,FO,FHEFF,GAIRX,GAIRY,ITAU,TDTS,
     *          NX1,NY1,NUMBER)
      CALL GROWTH(HEFF,AREA,HO,A22,FHEFF,FO,HCORR,HEFF4,OUT,NX1,NY1,GARE
     *A)
C     MUST CALL GROWTH ONLY AFTER CALLING ADVECTION
C
C     SUM OF TOTAL ICE IN THE BASIN... EXCLUDING OUTFLOW CELLS
C
      CALL XSUM(HEFF,THEFF,NX1,NY1)
C
C     THIS SECTION COMPUTE VARIOUS SUMS NECESSARY FOR INSURING
C     CONSERVATION PLUS MONITORING VARIOUS CONTRIBUTIONS TO ICE
C     CHANGES.
C
      GR = 0.0
      THEFF2 = 0.0
      FHSUM = 0.0
      GRSUM = 0.0
      ARSUM = 0.0
      FHE1 = 0.0
      DO 105 J = 1,24
      DO 105 I = 1,23
      HEFF(I,J,1)=HEFF(I,J,1)*OUT(I,J)
      AREA(I,J,1)=AREA(I,J,1)*OUT(I,J)
      FHEFF(I,J)=FHEFF(I,J)*OUT(I,J)
C
C     HDIFF CONTAINS THE TOTAL OPEN WATER GROWTH FOR THE BASIN
C
      HDIFF(I,J) = (1.0 - AREA(I,J,2)) * FO(I,J) * OUT(I,J) * DELTT
      GRSUM = GRSUM + HDIFF(I,J)
      THEFF2 = THEFF2 + HEFF(I,J,1)
      ARSUM = ARSUM + AREA(I,J,1)
      FHSUM = FHSUM + FHEFF(I,J)
  105 CONTINUE
      GRSUM1=GRSUM1+GRSUM
C
C     GRSUM1 CONTAINS THE NET OPEN WATER GROWTH
C
      FHSUM1=FHSUM1+FHSUM
C
C     FHSUM CONTAINS THE NET ICE GROWTH
C
      ARSUM1=ARSUM1+ARSUM
      TOUT1=THEFF-THEFF2-THEFF1
      THEFF=THEFF2
      TOUT=TOUT+TOUT1
C
C
C     OUTPUT SECTION..... PRINT ON SPECIFIED TIME STEPS
C
C
   65 CONTINUE
      IF(LSTEP .NE. 3) GO TO 547
    6 FORMAT(//)
```

```
 1    FORMAT(* *,*TIME STEP AND TOTAL THICKNESS ARE*,I10,10X,G20.12)
      PRINT 91,TOTS(1)
91    FORMAT(1H1,*T H I C K N E S S      THE DATE IS   *,A8)
      PRINT 1,TCOUNT,THEFF1
      PRINT 1,TCOUNT,THEFF
      PRINT 4
      PRINT 2,TOUT1,TOUT
      PRINT 4
 2    FORMAT(1X,*OUTFLOW FOR THIS TIME STEP    *,F10.4/1X,
     **NET OUTFLOW     *,F10.4)
 3    FORMAT(1X,*OPEN WATER GROWTH   *,F10.4/1X,
     **NET OPEN WATER GROWTH    *,F10.4)
 8    FORMAT(1X,*ICE GROWTH FOR THIS TIME STEP  *,F10.4/
     *1X,*NET ICE GROWTH     *,F10.4)
      CALL HSPLOT(HEFF,AREA,IDTS,ITAU,GROT,GRIT,NX,NY)
      CALL GROWDEC(HDIFF,FHEFF,GAREA,IDTG,ITAU,NX1,NY1)
      PRINT 4
96    CONTINUE
      PRINT 1,TCOUNT,THEFF
      PRINT 2,TOUT1,TOUT
      PRINT 3,FHSUM,FHSUM1
      PRINT 8,GRSUM,GRSUM1,ARSUM,ARSUM1
      PRINT 517
517   FORMAT(1H0,*OPEN WATER GROWTH*)
      CALL PRNT(HDIFF,24,24,1,1,13,24)
      CALL PRNT(HDIFF,24,24,1,14,26,24)
      PRINT 520
520   FORMAT(1H0,*UICE   AND   VICE*)
      CALL PRNT(UICE,27,27,3,1,13,27)
      CALL PRNT(UICE,27,27,3,14,26,27)
      CALL PRNT(VICE,27,27,3,1,13,27)
      CALL PRNT(VICE,27,27,3,14,26,27)
      PRINT 521
521   FORMAT(1H0,*HEFF*)
      CALL PRNT(HEFF,24,24,3,1,13,24)
      CALL PRNT(HEFF,24,24,3,14,26,24)
      PRINT 522
522   FORMAT(1H0,*AREA*)
      CALL PRNT(AREA,24,24,3,1,13,24)
      CALL PRNT(AREA,24,24,3,14,26,24)
      PRINT 527
527   FORMAT(1H0,*SH TERM*)
      CALL PRNT(FHEFF,24,24,1,1,13,24)
      CALL PRNT(FHEFF,24,24,1,14,26,24)
      PRINT 529
529   FORMAT(1H0,*SA TERM*)
      CALL PRNT(GAREA,24,24,1,1,13,24)
      CALL PRNT(GAREA,24,24,1,14,26,24)
      PRINT 537
537   FORMAT(1H0,*ICE TO BE MELTED TO MAINTAIN MASS BALANCE*)
      CALL PRNT(HCORR,24,24,1,1,13,24)
      CALL PRNT(HCORR,24,24,1,14,26,24)
      PRINT 541
541   FORMAT(1H0,*ICE STRENGTH    N/M*)
      CALL PRNT(PRESS,24,24,1,1,13,24)
      CALL PRNT(PRESS,24,24,1,14,26,24)
      PRINT 642
642   FORMAT(1H0,*DIVERGENCE FIELD     SEC-1*)
      CALL PRNT(DIV,24,24,1,1,13,24)
      CALL PRNT(DIV,24,24,1,14,26,24)
```

```
       547 CONTINUE
C
C          CALL ADJUST TO ESTIMATE THE ICE IN THE OUTFLOW CELLS
C
           CALL ADJUST(HEFF,AREA,OUT,HEFFM,NX,NY,NX1,NY1)
C
C          DETERMINE IF THE ITERATION PROCESS CONTINUES
C
           CALL SECOND(T2)
           TT = T2 - TNEW
           PRINT 9123,TT
      9123 FORMAT(1H0,'*********** TIME STEP TIME   ',F10.4)
           IF(ICOUNT .EQ. ITSTEP) GO TO 205
C
C          CHECK IF NEW INPUT DATA IS REQUIRED
C
           LSTEP = LSTEP + 1
           IF(LSTEP .NE. 4) GO TO 100
           LSTEP = 0
           READ(2,16) IDTG(1), IDTG(2), IDTG(3)
           CALL INITIAL(1,GAIRX,GRDI,GRDJ,IDTG,NUMREP,ITAU)
           CALL INITIAL(2,GAIRY,GRDI,GRDJ,IDTG,NUMREP,ITAU)
           DO 1111 J = 1,29
           DO 1111 I = 1,29
           GAIRX(I,J) = GAIRX(I,J) * 0.01
           GAIRY(I,J) = GAIRY(I,J) * 0.01
      1111 CONTINUE
C          VALUES BACK.
C
           GO TO 100
       205 CONTINUE
C
C          WRITE OUT THE SUM TOTALS  FOR RESTART
C
           WRITE(3,731) GRSUM1, ARSUM1, FHSUM1,TOUT1
       731 FORMAT(1X,4E12.6)
           CALL SECOND(TSTOP)
           TSTOP = TSTOP - TBEGIN
           CALL STATPRT(TSTOP)
           STOP 'END OF ICE MODEL'
           END
```

```
            SUBROUTINE ADVECT(UICEC,VICEC,HEFF,DIFF1,LAD,HEFFM,NX,NY,NX1,NY1)
            COMMON /TIME/ RELAXS,FORMS,ADVCTS,GRWTHS,HEATS,MFSHS,INITS
            DIMENSION HEFF(28,28,3),UICEC(27,27),VICEC(27,27)
           *, HEFFM(28,28)
            COMMON /STEP/ DELTAT, DELTAX, DELTAY, DELTA1, DELTA
            CALL SECOND(T1)
            NXM1 = NX - 1
            NYM1 = NY - 1
            LL = LAD
      C
      C NOW DECIDE IF BACKWARD EULER OR LEAPFROG
      C
            IF(LL.EQ.1) GO TO 100
      C
      C BACKWARD EULER
      C
            DELTT=DELTAT
            K3=2
            K2=2
            GO TO 101
      C
      C LEAPFROG
      C
        100 DELTT=DELTAT*2.0
            K3=3
            K2=2
        101 CONTINUE
      C
      C NOW REARRANGE HKS
      C
            DO 200 J = 1,28
            DO 200 I = 1,28
            HEFF(I,J,3)=HEFF(I,J,2)
            HEFF(I,J,2)=HEFF(I,J,1)
        200 CONTINUE
        202 CONTINUE
      C
      C NOW GO THROUGH STANDARD CONSERVATIVE ADVECTION
      C
            DELTX=DELTT/(4.0*DELTAX)
            DELTY=DELTT/(4.0*DELTAY)
            DO 210 J = 1,26
            DO 210 I = 1,26
            HEFF(I+1,J+1,1)=HEFF(I+1,J+1,K3)-DELTX*((HEFF(I+1,J+1,2)+HEFF
           *(I+2,J+1,2)) * (UICEC(I+1,J+1) + UICEC(I+1,J)) - (HEFF(I+1,J+1,2)
           * + HEFF(I,J+1,2)) * (UICEC(I,J+1) + UICEC(I,J))) - DELTY *
           * ((HEFF(I+1,J+1,2) + HEFF(I+1,J+2,2)) * (VICEC(I,J+1) +
           * VICEC(I+1,J+1)) - (HEFF(I+1,J+1,2) + HEFF(I+1,J,2)) *
           * (VICEC(I,J) + VICEC(I+1,J)))
        210 CONTINUE
      C
      C NOW DECIDE IF DONE
      C
            IF(LL.EQ.2) GO TO 99
            IF (LL.EQ.3) GO TO 99
            GO TO 102
         99 CONTINUE
      C  NOW FIX UP H(I,J,2)
            DO 98 J = 1,28
            DO 98 I = 1,28
            HEFF(I,J,2)=HEFF(I,J,3)
      88    CONTINUE
            GO TO 102
      99    CONTINUE
     C 100. DO BACKWARD EULER CORRECTION
            DO 220 J=1,NY1
            DO 220 I=1,NX1
            HEFF(I,J,3)=HEFF(I,J,2)
            HEFF(I,J,2)=0.5*(HEFF(I,J,1)+HEFF(I,J,2))
      220   CONTINUE
            LL=3
            K3=3
            GO TO 202
      102   CONTINUE
     C NOW DO DIFFUSION ON H(I,J,K3)
            DO 240 K2=1,2
     C      GO TO (241,242),K2
            IF(K2.EQ.1) GO TO 241
            IF(K2.EQ.2) GO TO 242
      241   CONTINUE
            CALL DIFFUS(UICE,VICE,HEFF,DIFF1,DELTT,HEFFM,NX,NY,NX1,NY1)
            GO TO 243
      242   CONTINUE
            DIFF2=-(DELTAX**2)/DELTT
            CALL DIFFUS(UICE,VICE,HEFF,DIFF2,DELTT,HEFFM,NX,NY,NX1,NY1)
      243   CONTINUE
            DO 330 J = 1,24
            DO 330 I = 1,24
            HEFF(I,J,1)=(HEFF(I,J,1)+HEFF(I,J,3))*HEFFM(I,J)
      330   CONTINUE
      240   CONTINUE
            CALL SECOND(T2)
            ADVCTS = ADVCTS + (T2 - T1)
            RETURN
            END
    NO ERRORS
```

```
FTN 1.5.1 CYCLE FTN1556  BUILT 02/03/81 23.27    SOURCE LISTING
            SUBROUTINE DIFFUS(UICE,VICE,HEFF,DIFF1,DELTT,HEFFM,NX,NY,NX1,NY1)
      C   SPACER
            DIMENSION HEFF(28,28,3),UICE(27,27,3),VICE(27,27,3),
          *      HEFF1(28,28),HEFFM(28,28)
            COMMON /PRESS/ PRESS(28,28)
            COMMON /STEP/ DELTAT,DELTAX,DELTAY,DELTA1,DELTA
      C
      C SUBROUTINE DIFFUSES HEFF,MULTIPLIES BY DELT, AND PUTS RESULTS IN HEFF
      C   NOW ZERO OUT HEFF1
      C
            DO 210 J = 1,28
            DO 210 I = 1,28
            HEFF1(I,J)=0.0
        210 CONTINUE
      C NOW DO DIFFUSION
            DELTXX=DELTT*DIFF1/(DELTAX**2)
            DELTYY=DELTT*DIFF1/(DELTAY**2)
            DO 220 J = 2,27
            DO 220 I = 2,27
            HEFF1(I,J)=DELTXX*((HEFF(I+1,J,3)-HEFF(I,J,3))*HEFFM(I+1,J)
          *    -(HEFF(I,J,3)-HEFF(I-1,J,3))*HEFFM(I-1,J))
          *   +DELTYY*((HEFF(I,J+1,3)-HEFF(I,J,3))*HEFFM(I,J+1)
          *   -(HEFF(I,J,3)-HEFF(I,J-1,3))*HEFFM(I,J-1))
        220 CONTINUE
            DO 250 J = 1,28
            DO 250 I = 1,28
            HEFF(I,J,3)=HEFF1(I,J)
        250 CONTINUE
            RETURN
            END
   NO ERRORS
```

```
'TRAN 1.5.1 CYCLE FTN1566  BUILT 04/03/81 23.27   SOURCE LISTING
                   SUBROUTINE XSUM(HEFF,S1,NX1,NY1)
          C
          C PROGRAM SUMS UP VECTOR
          C
                   DIMENSION HEFF(24,24,3)
                   S1=0.0
                   DO 100 J = 1,NY1
                   DO 100 I = 1,NX1
                   S1 = S1 + HEFF(I,J,1)
            100    CONTINUE
                   RETURN
                   END
    NO ERRORS
```

```
TRAN 1.5.1 CYCLE FTN1536  BUILT 09/03/81 23.27   SOURCE LISTING
                 SUBROUTINE MEAN(HEFF,HMEAN,NX,NY,OUT)
        C
        C SUBROUTINE FINDS MEAN HEFF AT OUTFLOW PTS OF VALUES AROUND
        C
                 DIMENSION HEFF(28,28,3), HMEAN(28,28), OUT(28,28)
                 DO 101 J=2,NY
                 DO 101 I=2,NX
                 HMEAN(I,J)=(HEFF(I+1,J,1)*OUT(I+1,J)+HEFF(I+1,J+1,1)*OUT(I+1,J+1)
                * +HEFF(I+1,J-1,1)*OUT(I+1,J-1)+HEFF(I,J+1,1)*OUT(I,J+1)
                * +HEFF(I,J-1,1)*OUT(I,J-1)+HEFF(I-1,J,1)*OUT(I-1,J)
                *  +HEFF(I-1,J+1,1)*OUT(I-1,J+1)+HEFF(I-1,J-1,1)*OUT(I-1,J-1)
                *)/(OUT(I+1,J)+OUT(I+1,J+1)+OUT(I+1,J-1)+OUT(I,J+1)+OUT(I,J-1)
                *  +OUT(I-1,J)+OUT(I-1,J+1)+OUT(I-1,J-1)+.00001)
            101  CONTINUE
                 RETURN
                 END
      NO ERRORS
```

```
TRAN 1.5.1 CYCLE FTN1596  BUILT 04/03/81 23 27   SOURCE LISTING
            SUBROUTINE OCEAN(SWATX,SWATY,NX,NY,GRDI,GRDJ,IVM)
            DIMENSION IVM(27,27), WATD(27,27), WATE(27,27)
            DIMENSION SWATX(27,27), SWATY(27,27), GRDI(29,29), GRDJ(29,29)
            COMMON /STEP/ DELTAT, DELTAX, DELTAY, DELTAI, DELTA
            DIMENSION WD(10,10), WE(10,10)
      C
      C
      C
            DATA ((WE(I,J),I=1,10),J=1,10) /
           1   0.,   0.,   0.,   6.,  11.,   0.,   0.,   0.,   0.,   0.,
           2   0.,   0.,   7.,   3.,   4.,   4.,   3.,   0.,   0.,   0.,
           3   0.,   7.,   1.,   3.,   4.,   4.,   1.,   4.,   0.,   0.,
           4   1.,   4.,   4.,   1.,   3.,   0.,  10.,   3.,   0.,   0.,
           5   5.,   7.,   3.,   1.,   1.,   1.,   4.,   0.,   0.,   1.,
           6   0.,   4.,   3.,   1.,   1.,   1.,   3.,   0.,   0.,   0.,
           7   0.,   4.,   0.,   1.,   1.,   1.,   3.,   0.,  22.,   0.,
           8  11.,   4.,   4.,   1.,   1.,   4.,   1.,   0.,  11.,  22.,
           9   4.,   7.,   4.,   0.,   3.,  11.,   0.,   0.,   1.,   4.,
           1   1.,  11.,   4.,   7.,  11.,  21.,   1.,   0.,   1.,   0. /
            DO 25 J = 1,27
            DO 25 I = 1,27
            SWATX(I,J) = 0.0
            SWATY(I,J) = 0.0
            WATD(I,J) = 0.0
            WATE(I,J) = 0.0
   25       CONTINUE
            DO 7 J = 2,26
            READ (7,50) (WATD(I,J),I = 2,26)
    7       CONTINUE
   50       FORMAT(25F3.0)
            RJ = 1.0
            DO 15 J = 2,26
            RK = 1.0
            DO 10 I = 2,26
            WATD(I,J) = WATD(I,J) * 10.0
            RI = 1.0 + (RK - 1.0) * DELTAI
            RI = RI + 0.5
            IG = AINT(RI)
            RR = RJ + 0.5
            JG = AINT(RR)
            WATE(I,J) = WE(IG,JG) / 100.0
            RK = RK + 1
   10       CONTINUE
            RJ = RJ + DELTAI
            RR = RJ + 0.5
            JG = AINT(RR)
   15       CONTINUE
            PRINT 900
  900       FORMAT(1X,'WATD AND WATE')
            DO 30 J = 1,NY
            DO 30 I = 1,NX
            X = GRDI(I,J) - 31.0
            Y = GRDJ(I,J) - 31.0
            CALL DIFFUV(SWATX(I,J),SWATY(I,J),WATD(I,J),WATE(I,J),X,Y)
   30       CONTINUE
            RETURN
            END
     NO ERRORS
```

```
FTRAN 1.5.1 CYCLE FTN1535  BUILT 03/03/81 23 27   SOURCE LISTING
                 SUBROUTINE BNDRY(HEFFM,UVM,OUT,NX,NY,NX1,NY1)
          C
          C SUBROUTINE SETS UP BOUNDARY MASK
          C
                 DIMENSION HEFFM(28,28), UVM(27,27), OUT(28,28)
          C
          C READ IN VELOCITY MASK
          C
                 DO 10 J = 1,NY
                 READ (7,50) (UVM(I,J),I=1,NX)
            10   CONTINUE
            50   FORMAT(27F2.0)
                 DO 20 J = 1,NY1
                 READ (7,55) (HEFFM(I,J),I=1,NX1)
            20   CONTINUE
            55   FORMAT(28F2.0)
                 DO 30 J = 1,NY1
                 READ (7,55) (OUT(I,J),I=1,NX1)
            30   CONTINUE
                 RETURN
                 END
      NO ERRORS
```

```
RTRAN 1.5.1 CYCLE FTN1586  BUILT 09/03/81 23.27   SOURCE LISTING
              SUBROUTINE RELAX(UICE,VICE,ETA,ZETA,DRAGS,DRAGA,AMASS,UVM,
             *ERROR,THETA,UICEC,VICEC,HEFFM,NX,NY,NX1,NY1)
              DIMENSION   UICE(27,27,3), VICE(27,27,3), ETA(28,28)
             *,              ZETA(28,28), DRAGS(28,28), DRAGA(28,28)
             *,             FXETA(4),FYETA(4),UICEC(27,27),VICEC(27,27)
             *,              FYZETA(4), FXZETA(4), AMASS(27,27)
             *,              COEFF(27,27),FXM(27,27),FYM(27,27)
             *,              HEFFM(28,28), UVM(27,27)
             *,        FXE(27,27,4), FYE(27,27,4), FYZ(27,27,4), FXZ(27,27,4)
              COMMON /FORCE/ FORCEX(27,27),FORCEY(27,27)
              COMMON /STEP/ DELTAT, DELTAX, DELTAY, DELTA1, DELTA
              COMMON /PRESS/ PRESS(28,28)
              COMMON /TIME/ RELAXS,FORMS,ADVCTS,GRWTHS,HEATS,MESHS,UNITS
              CALL SECOND(T1)
              ICOUNT=0
              NXM1 = NX - 1
              NYM1 = NY - 1
              WFA = 1.7
              DELIN =1.0/DELTAX
              DELIN2=0.5/(DELTAX**2)
              K=1
        C
        C MUST UPDATE HEFF BEFORE CALLING RELAX
        C FIRST SET U(2)=U(1)
        C
              DO 99 J=1,NY
              DO 99 I=1,NX
        C
        C NOW MAKE SURE BDRY PTS ARE EQUAL TO ZERO
        C
              UICE(I,J,2)=UICE(I,J,1)
              VICE(I,J,2)=VICE(I,J,1)
              UICE(I,J,1)=UICE(I,J,3)*UVM(I,J)
              VICE(I,J,1)=VICE(I,J,3)*UVM(I,J)
           99 CONTINUE
        C
        C NOW SET UP COEFFICIENTS OF DIAGONAL COMPONENTS
        C
              DO 102 J = 2,26
              DO 102 I = 2,26
              C = AMASS(I,J)/DELTAT + 2.0 * THETA * (0.5 * DRAGS(I,J)
             *+2.0*((ETA(I,J)+ETA(I+1,J)+ETA(I,J+1)+ETA(I+1,J+1))
             *+.5*(ZETA(I,J)+ZETA(I+1,J)+ZETA(I,J+1)+ZETA(I+1,J+1))
             *  )/(4.0*(DELTAX**2)))
              COEFF(I,J) = 1.0/C
          102 CONTINUE
        C
        C NOW CALCULATE ALL FUNCTIONS OF PREVIOUS U AND V VALUES
        C
              TTHETA=2.0*(1.0-THETA)
              DO 111 J=2,NYM1
              DO 111 I=2,NXM1
              CALL FELLIP(UICE,VICE,ETA,FXETA,I,J,2)
              CALL FELLIP(UICE,VICE,ZETA,FXZETA,I,J,2)
              CALL FELLIP(VICE,UICE,ETA,FYETA,I,J,2)
              CALL FELLIP(VICE,UICE,ZETA,FYZETA,I,J,2)
              FX0 = 0.5 * (FXETA(1)+FXZETA(1)+FXETA(2)+FXETA(3)+FXZETA(4)-FXETA(
             *4))
              FX0=TTHETA*FX0
              FX1=(AMASS(I,J)/DELTAT-TTHETA*0.5*DRAGS(I,J))*UICE(I,J,2)
             FX2=TTHETA*0.5*DRAGA(I,J)*VICE(I,J,2)
             FY0 = 0.5 * (FYETA(1)+FYETA(2)+FYZETA(2)+FYZETA(3)-FYETA(3) +
            *FYETA(4))
             FY0=FY0*TTHETA
             FY1=(AMASS(I,J)/DELTAT-TTHETA*0.5*DRAGS(I,J))*VICE(I,J,2)
             FY2=-TTHETA*0.5*DRAGA(I,J)*UICE(I,J,2)
             FXC=AMASS(I,J)*0.5*TTHETA*
            *   (UICEC(I,J)*(UICE(I+1,J,2)-UICE(I-1,J,2))
            *   +VICEC(I,J)*(UICE(I,J+1,2)-UICE(I,J-1,2)))/(2.0*DELTAX)
             FXM(I,J)=FX0+FX1+FX2+FORCEX(I,J)+FXC
             FYC=AMASS(I,J)*0.5*TTHETA*
            *    (UICEC(I,J)*(VICE(I+1,J,2)-VICE(I-1,J,2))
            *   +VICEC(I,J)*(VICE(I,J+1,2)-VICE(I,J-1,2)))/(2.0*DELTAY)
             FYM(I,J)=FY0+FY1+FY2+FORCEY(I,J)+FYC
       111   CONTINUE
     C
     C NOW SET U(3)=U(1)
     C
       100   CONTINUE
             DO 101 J=1,NY
             DO 101 I=1,NX
             UICE(I,J,3)=UICE(I,J,1)
             VICE(I,J,3)=VICE(I,J,1)
       101   CONTINUE
     C
     C NOW BEGIN SWEEP
     C
             CALL FELLD1(UICE,VICE,ETA,FXE,1,DELTN2)
             CALL FELLD1(UICE,VICE,ZETA,FXZ,1,DELTN2)
             CALL FELLD1(VICE,UICE,ETA,FYE,1,DELTN2)
             CALL FELLD1(VICE,UICE,ZETA,FYZ,1,DELTN2)
             DO 103 J = 2,25
             DO 103 I = 2,25
             K=1
             FXETA(1) = FXE(I,J,1) + DELTN2 *
            *(UICE(I-1,J,K)*( ETA(I,J+1)+ ETA(I,J)))
             FXETA(2) = FXE(I,J,2) + DELTN2 *
            *(UICE(I,J-1,K)*(ETA(I,J)+ETA(I+1,J)))
             FXETA(3) = FXE(I,J,3) + DELTN2 * 0.5 *
            *(VICE(I-1,J-1,K)*ETA(I,J)+VICE(I,J-1,K)*
            *(-ETA(I,J)+ETA(I+1,J))-VICE(I+1,J-1,K)*ETA(I+1,J)
            *+VICE(I-1,J,K)*(-ETA(I,J+1)+ETA(I,J))
            *-VICE(I-1,J+1,K)*ETA(I,J+1)))
             FXETA(4) = FXE(I,J,4) +DELTN2 * 0.5 *
            *(VICE(I-1,J-1,K)*ETA(I,J)+VICE(I,J-1,K)*
            *(-ETA(I+1,J)+ETA(I,J))-VICE(I+1,J-1,K)*ETA(I+1,J)
            *+VICE(I-1,J,K)*(ETA(I,J+1)-ETA(I,J))
            *-VICE(I-1,J+1,K)*ETA(I,J+1)))
     C
     C
             FYETA(1) = FYE(I,J,1) + DELTN2 * (VICE(I-1,J,K)
            *  * (ETA(I,J+1) + ETA(I,J)))
             FYETA(2)= FYE(I,J,2) + DELTN2 *
            *(VICE(I,J-1,K)*(ETA(I,J)+ETA(I+1,J)))
             FYETA(3) = FYE(I,J,3) + 0.5 * DELTN2 *
            *(UICE(I-1,J-1,K)*ETA(I,J)+UICE(I,J-1,K)*
            *(-ETA(I,J)+ETA(I+1,J))-UICE(I+1,J-1,K)*ETA(I+1,J)
            *+UICE(I-1,J,K)*(-ETA(I,J+1)+ETA(I,J))
            *-UICE(I-1,J+1,K)*ETA(I,J+1)))
             FYETA(4) = FYE(I,J,4) + 0.5 * DELTN2 *
     *(UICE(I-1,J-1,K)*ETA(I,J)+UICE(I,J-1,K)*
     *(-ETA(I+1,J)+ETA(I,J))-UICE(I+1,J-1,K)*ETA(I+1,J)
     *+UICE(I-1,J,K)*(ETA(I,J+1)-ETA(I,J))
     *-UICE(I-1,J+1,K)*ETA(I,J+1))
C
C
      FXZETA(1) = FXZ(I,J,1) + DELTN2 *
     *(UICE(I-1,J,K)*(ZETA(I,J+1)+ZETA(I,J)))
      FXZETA(4) = FXZ(I,J,4) + DELTN2 * 0.5 *
     *(VICE(I-1,J-1,K)*ZETA(I,J)+VICE(I,J-1,K)*
     *(-ZETA(I+1,J)+ZETA(I,J))-VICE(I+1,J-1,K)*ZETA(I+1,J)
     *+VICE(I-1,J,K)*(ZETA(I,J+1)-ZETA(I,J))
     *-VICE(I-1,J+1,K)*ZETA(I,J+1))
C
C
      FYZETA(2) = FYZ(I,J,2) + DELTN2 *
     *(VICE(I,J-1,K)*(ZETA(I,J)+ZETA(I+1,J)))
      FYZETA(3) = FYZ(I,J,3) + DELTN2 * 0.5 *
     *(UICE(I-1,J-1,K)*ZETA(I,J)+UICE(I,J-1,K)*
     *(-ZETA(I,J)+ZETA(I+1,J))-UICE(I+1,J-1,K)*ZETA(I+1,J)
     *+UICE(I-1,J,K)*(-ZETA(I,J+1)+ZETA(I,J))
     *-UICE(I-1,J+1,K)*ZETA(I,J+1))
C
C
C
      FX3 = 0.5 * (FXETA(1) + FXZETA(1) + FXETA(2) + FXETA(3) +FXZETA(4)
     * - FXETA(4))
      FX3=FX3*2.0*THETA
      FXCP=AMASS(I,J)*THETA*
     *   (UICEP(I,J)*(UICE(I+1,J,1)-UICE(I-1,J,1))
     *+VICEP(I,J)*(UICE(I,J+1,1)-UICE(I,J-1,1)))*0.5*DELTN
      FX3=FX3+FXCP
      FY3 = 0.5 * (FYETA(1) + FYETA(2) + FYZETA(2) + FYZETA(3)
     * - FYETA(3) + FYETA(4))
      FY3=FY3*2.0*THETA
      FYCP=AMASS(I,J)*THETA*
     *   (UICEP(I,J)*(VICE(I+1,J,1)-VICE(I-1,J,1))
     *+VICEP(I,J)*(VICE(I,J+1,1)-VICE(I,J-1,1)))*0.5*DELTN
      FY3=FY3+FYCP
      FL11=0.5*DRAGA(I,J)*COEFI(I,J)
      FL11=FL11*2.0*THETA
      F11=(FX4(I,J)+FX3)*COEFI(I,J)
      F22=(FY4(I,J)+FY3)*COEFI(I,J)
      FL11S=1.0+FL11**2
      FL11ST=1.0/FL11S
      UICOR=((F11+FL11*F22)*FL11ST)*IVM(I,J)
      VICOR=((F22-FL11*F11)*FL11ST)*IVM(I,J)
      UICE( I,J,1 ) = UICE ( I,J,1 ) + WFA*( UICOR-UICE(I,J,1))
      VICE( I,J,1 ) = VICE ( I,J,1 ) + WFA*( VICOR-VICE(I,J,1))
  103 CONTINUE
      ICOUNT=ICOUNT+1
      IF(ICOUNT .GT. 1300) GO TO 201
      IF(ICOUNT .GT. 100) WFA = 1.0
C
C NOW CHECK MAX ERROR
C FORM ERROR MATRIX
C
C
      S1 = 0.0
      S2 = 0.0
```

```
            DO 104 J = 1,NY
            DO 104 I=1,NX
            UERR = UICE(I,J,1) - UICE(I,J,3)
            VERR = VICE(I,J,1) - VICE(I,J,3)
            S1 = AMAX1( ABS(UERR),S1 )
            S2 = AMAX1( ABS(VERR),S2 )
      104   CONTINUE
            S1 = AMAX1( S1,S2 )
            IF(S1.LT.ERROR) GO TO 200
            GO TO 100
      201   CONTINUE
            PRINT 11
       11   FORMAT(1X,'NO CONVERGENCE AFTER 800 ITERATIONS')
    C NOW END
      200   CONTINUE
    C       PRINT 1,ICOUNT
            PRINT 12,S1
            PRINT 1,ICOUNT
       12   FORMAT(1X,'MAX ERROR AND U AND V POWER    ',3F12.5)
        1   FORMAT(1X,'NUMBER OF ITERATIONS ARE    ',I20)
            CALL SECOND(T2)
            RELAXS = RELAXS + (T2 - T1)
            RETURN
            END
  NO ERRORS
```

```
RTRAN 1.5.1 CYCLE FTN1536  BUILT 08/03/81 23 27   SOURCE LISTING
 00001          SUBROUTINE FELLIP(UICE,VICE,ETA,F,I,J,K)
        C SPACER
 00002          DIMENSION UICE(27,27,3),VICE(27,27,3),ETA(28,28),F(4)
 00003          COMMON /STEP/ DELTAT, DELTAX, DELTAY, DELTA1, DELTA
 00004          S1=.5/(DELTAX**2)
 00005          F(1)=S1*(UICE(I+1,J,K)*(ETA(I+1,J+1)+ETA(I+1,J))
               *-UICE(I,J,K)*(ETA(I+1,J+1)+ETA(I,J)+ETA(I+1,J)+ETA(I,J+1))
               *+ UICE(I-1,J,K)*(ETA(I,J+1)+ETA(I,J)))
 00006          F(2)=S1*(UICE(I,J+1,K)*(ETA(I+1,J+1)+ETA(I,J+1))
               *-UICE(I, J,K)*(ETA(I+1,J+1)+ETA(I,J)+ETA(I+1,J)+ETA(I,J+1))
               *+UICE(I, J-1,K)*(ETA(I,J)+ETA(I+1,J)))
 00007          F(3)=S1*(VICE(I-1,J-1,K)*ETA(I,J)+VICE(I,J-1,K)*(-ETA(I,J)
               *+ETA(I+1,J))-VICE(I+1,J-1,K)*ETA(I+1,J)+VICE(I-1,J,K)*(-ETA(I, J+1)
               *+ETA(I, J))   +VICE(I,J,K)*(-ETA(I,J)-ETA(I+1,J+1)+ETA(I+1,J)
               *+ETA(I, J+1)))
 00008          F(3)=F(3)+S1*(VICE(I+1,J,K)*(-ETA(I+1,J)+ETA(I+1,J+1))
               *-VICE(I-1,J+1,K)*ETA(I,J+1)
               *+VICE(I,J+1,K)*(-ETA(I+1,J+1)+ETA(I,J+1))
               *+VICE(I+1,J+1,K)*ETA(I+1,J+1))
 00009          F(4)=S1*(VICE(I-1, J-1,K)*ETA(I,J)+VICE(I,J-1,K)*(-ETA(I+1,J)
               *+ETA(I, J))-VICE(I+1,J-1,K)*ETA(I+1,J)+VICE(I-1, J,K)*(ETA(I,J+1)
               *-ETA(I, J))+VICE(I,J,K)*(ETA(I, J+1)+ETA(I+1,J)-ETA(I,J)-ETA(I+1,
               * J+1)))
 00010          F(4)=F(4)+S1*(VICE(I+1,J,K)*(ETA(I+1,J)-ETA(I+1,J+1))
               *-VICE(I-1,J+1,K)*ETA(I,J+1)
               *+VICE(I,J+1,K)*(ETA(I+1,J+1)-ETA(I,J+1))
               *+VICE(I+1,J+1,K)*ETA(I+1,J+1))
 00011          F(3)=F(3)*.5
 00012          F(4)=F(4)*.5
 00013          RETURN
 00014          END
     NO ERRORS
```

```
FTRAN 1.5.1 CYCLE FTN1536  BUILT 08/03/81 23 27   SOURCE LISTING                  2
 00001           SUBROUTINE AVG(ARRAY,N)
 00002           DIMENSION ARRAY(20,20)
 00003           NM1 = N - 1
 00004           DO 10 I = 1,NM1
 00005           DO 10 J = 1,NM1
 00006           ARRAY(I,J) = (ARRAY(I,J) + ARRAY(I,J+1) + ARRAY(I+1,J)
                *  + ARRAY(I+1,J+1)) / 4.0
 00007     10    CONTINUE
 00008           RETURN
 00009           END
     NO ERRORS
```

```
FTRAN 1.5.1 CYCLE FTN1534  BUILT 09/03/81 23 27    SOURCE LISTING
            SUBROUTINE HEAT(GRDI,GRDJ,HEFF,AREA,FO,FHEFF,GSTRX,GSTRY,ITAN,
           *                  IDTS,NXI,NYI,NUMBER)
            CHARACTER*8 ITAN,IDTS
            COMMON /ITYPE/ RELAXS,FORCES,ADVCTS,GROTHS,HEATS,FRESHS,LIMIT
            COMMON /PROD/ FSH(24,24)
            DIMENSION TICE(23,24), TMIX(23,23), TAIR(24,24), QA(24,24),
           *          FLO(24,23), GRDI(24,24), GRDJ(24,23), GSTRX(24,24),
           *          GSTRY(24,24), PS(23,24), FS(23,23),FHEFF(24,24),
           *          FO(23,24), AREA(23,24,3), HEFF(24,24,3),IDTS(3)
           *,         IS(24,24)
            CALL SECOND(T1)
      C
      C
            DO 10 I = 1,NXI
            DO 10 J = 1,NYI
      C
      C
            IS(I,J) = SQRT(GSTRX(I,J) * * 2 + GSTRY(I,J) * * 2)
        10  CONTINUE
            CALL INITIAL(3,TAIR,GRDI,GRDJ,IDTS,NUMBER,0)
            CALL AVG(TAIR,NUMBER)
      C
      C
      C
            DO 15 J = 1,24
            DO 15 I = 1,24
            TAIR(I,J) = TAIR(I,J) + 273.0
        15  CONTINUE
            CALL INITIAL(4,PS,GRDI,GRDJ,IDTS,NUMBER,0)
            CALL AVG(PS,NUMBER)
            CALL INITIAL(5,FS,GRDI,GRDJ,IDTS,NUMBER,0)
            CALL AVG(FS,NUMBER)
            DO 20 J = 1,23
            DO 20 I = 1,23
            QA(I,J) = (0.622 * FS(I,J)) / (PS(I,J) - FS(I,J))
        20  CONTINUE
            CALL INITIAL(6,FSH,GRDI,GRDJ,IDTS,NUMBER,0)
            CALL AVG(FSH,NUMBER)
            CALL INITIAL(7,PS,GRDI,GRDJ,IDTS,NUMBER,0)
            CALL AVG(PS,NUMBER)
            CALL INITIAL(8,FS,GRDI,GRDJ,IDTS,NUMBER,0)
            CALL AVG(FS,NUMBER)
            RINV = 1./86.
            DO 30 J = 1,23
            DO 30 I = 1,23
            PS(I,J) = PS(I,J) * RINV
            FS(I,J) = FS(I,J) * RINV
            FSH(I,J) = FSH(I,J) * RINV
            FLO(I,J) = PS(I,J) - FS(I,J) + FSH(I,J)
        30  CONTINUE
      C
      C
      C
            HMIX = 1.0 / 30.0
            DO 200 J = 1,24
            DO 200 I = 1,24
            AREA(I,J,2) = AMAX1(0.15,AREA(I,J,2))
       200  CONTINUE
            DO 201 J = 1,24
            DO 201 I = 1,24
```

```
                 TMIX(I, 1) = 271.2
                 TICE(I, 1) = 273.0
          201    CONTINUE
                 KOPEN = -1
                 CALL RIDGET(HEFF,FD,KOPEN,NX1,NY1,JS,TICE,TMIX,TAIR,DA,FLO)
                 KOPEN = 2
                 CALL RIDGET(HEFF,FHEFF,KOPEN,NX1,NY1,JS,TICE,TMIX,TAIR,DA,FLO)
                 DO 1047 I = 2,NX1
                 DO 1047 J = 2,NY1
                 FHEFF(I,J) = FHEFF(I,J) * AREA(I,J,2) + (1.0 - AREA(I,J,2)) * FD(I
               *,J)
          1047   CONTINUE
                 CALL SECOND(T2)
                 HEATS = HEATS + (T2 - T1)
                 RETURN
                 END
     NO ERRORS
```

```
      SUBROUTINE BUDGET(HEFF,FICE,KOPEN,NX1,NY1,UG,TICE,TMIX,TAIR,
     *                  QA,FLO)
      DIMENSION TICE(28,28), HEFF(28,28,3), FICE(28,28), TMIX(28,28),
     *          TAIR(28,28), QA(28,28), FLO(28,28),UG(28,28)
      COMMON /RAD/ FSH(28,28)
      QSI = 0.622/1013.0
      C1=2.773820E-6
      C2=-2.6913393E-03
      C3=0.97920849
      C4=-158.63779
      C5=9653.1925
      QQ=1.0E-06/302.0
      FW=-2.0
      TB=271.2
      IMAX=3
      D1=2.242
      D1W = 5.6875E+03
      D1I = 6.4475E+03
      D3 = 5.5E-08
      TMELT=273.16
      TMELTP=273.159
      IF(KOPEN.GT.0) GO TO 51
      PRINT 1000
 1000 FORMAT(1X,'COMPUTING THIN ICE GROWTH RATE')
      DO 101 J = 1,28
      DO 101 I = 1,28
      ALB =0.1
      A1 = 0.9 * FSH(I,J) + FLO(I,J) + D1 * UG(I,J) * TAIR(I,J) +
     *     D1W * UG(I,J) * QA(I,J)
      B = QSI * 6.11 * EXP(17.2694 * (TMIX(I,J) - TMELT) /
     *    (TMIX(I,J) - TMELT + 237.3))
      A2 = - D1 * UG(I,J) * TMIX(I,J) - D1W * UG(I,J) * B - D3 *
     *     (TMIX(I,J) * * 4)
      FICE(I,J) = QQ * (FW - A1 - A2)
  101 CONTINUE
      RETURN
   51 CONTINUE
      PRINT 1005
 1005 FORMAT(1X,'COMPUTING THICK ICE GROWTH RATE')
      ITER =0
   60 CONTINUE
      DO 104 J = 1,28
      DO 104 I = 1,28
      HEFF(I,J,2) = AMAX1(HEFF(I,J,2),0.05)
  104 CONTINUE
      DO 105 J = 1,28
      DO 105 I = 1,28
      ALB = 0.75
      IF(TICE(I,J) .GT. TMELTP) ALB = 0.616
      A1 = (1.0 - ALB) * FSH(I,J) + FLO(I,J) + D1 * UG(I,J) * TAIR(I,J)
     *     + D1I * UG(I,J) * QA(I,J)
      B = QSI * (C1 * TICE(I,J) * * 4 + C2 * TICE(I,J) * * 3 + C3 *
     *    TICE(I,J) * * 2 + C4 * TICE(I,J) + C5)
      A2 = -D1 * UG(I,J) * TICE(I,J) - D1I * UG(I,J) * B - D3 *
     *     (TICE(I,J) * * 4)
      B = 2.1656 / HEFF(I,J,2)
      A3 = 4.0 * D3 * (TICE(I,J) * * 3) + B + D1 * UG(I,J)
      B = B * (TB - TICE(I,J))
      TICE(I,J) = TICE(I,J) + (A1 + A2 + B) / A3
      FICE(I,J) = QQ * (FW - A1 - A2)
   105  CONTINUE
        ITER=ITER+1
        DO 107 J=1.NY1
        DO 107 I=1.NX1
        TICE(I.J)=AMIN1(TICE(I.J).TMELT)
   107  CONTINUE
        IF(ITER .GT. IMAX) GO TO 52
        GO TO 50
    52  CONTINUE
        RETURN
        END
    NO ERRORS
```

```
C
            FORCEX(I,J) = FORCEX(I,J) + DWATN(I,J) * (COSWAT * SWATX(I,J) -
     * SINWAT * SWATY( I,J))
            FORCEY(I,J) = FORCEY(I,J) + DWATN(I,J) * (SINWAT * SWATY(I,J)
     * +COSWAT * SWATY( I,J ))
  107       CONTINUE
C
C     NOW AT IN TILT
C
            DO 109 J = 1,27
            DO 109 I = 1,27
            FORCEX(I,J) = FORCEX(I,J) - COR(I,J) * SWATY(I,J)
            FORCEY(I,J) = FORCEY(I,J) + COR(I,J) * SWATX(I,J)
  109       CONTINUE
C
C           STORE THE INPUT DATA
C
C NOW SET UP THE PRESSURE AND VISCOSITIES
C FIRST SET UP CONSTANTS
C           5.787D-08 IS 0.5 PER CENT PER DAY STRAIN RATE
C NOW SET UP VALUES
            DO 115 J = 1,27
            DO 115 I = 1,27
            PRESS(I,J) = 5.0E+03 * HEFF(I,J,1) * EXP(-20.0 * (1.
     * - AREA(I,J,1))))
  115       CONTINUE
 1000       CONTINUE
            CALL PLAST( VISC,VISC,PRESS,ETA,ZETA,ECCEN,
     *  HEFFM,NX1,NY1,DIM)
C NOW SET VISCOSITIES AND PRESSURE EQUAL TO ZERO AT OUTFLOW PTS
            DO 106 J=1,NY1
            DO 106 I=1,NX1
            ETA(I,J)=ETA(I,J)*OUT(I,J)
            ZETA(I,J)=ZETA(I,J)*OUT(I,J)
  106       CONTINUE
C NOW CALCULATE PRESSURE FORCE AND ADD TO EXTERNAL FORCE
            DO 117 J = 1,27
            DO 117 I = 1,27
            FORCEX(I,J)=FORCEX(I,J)-(0.25/DELTAX)*
     *((PRESS(I+1,J) * OUT(I+1,J)) + (PRESS(I+1,J+1) * OUT(I+1,J+1))
     * - (PRESS(I,J) * OUT(I,J)) - (PRESS(I,J+1) * OUT(I,J+1)))
            FORCEY(I,J)=FORCEY(I,J)-(0.25/DELTAY)*
     * ((PRESS(I,J+1) * OUT(I,J+1)) + (PRESS(I+1,J+1) * OUT(I+1,J+1))
     * - (PRESS(I,J) * OUT(I,J)) - (PRESS(I+1,J) * OUT(I+1,J)))
C NOW PUT IN MINIMAL MASS FOR TIME STEPPING CALCULATIONS
  117       CONTINUE
            CALL SECOND(T2)
            FORMS = FORMS + (T2 - T1)
            RETURN
            END
   NO ERRORS
```

```
TRAN 1.5.1 CYCLE FTN1585  BUILT 02/03/81 23 27   SOURCE LISTING
               SUBROUTINE FORM(UICE,VICE,ETA,ZETA,AMASS,GAIRX,GAIRY,GWATX,GWATY,
              *   DRAGS,DRAGA,OUT,HEFFM,NX,NY,NX1,NY1,DIV,HEFF,AREA)
          C
          C PROGRAM FORMS BASIC INPUT PARAMETERS FOR RELAXATION
          C
               DIMENSION  UICE(27,27,3),VICE(27,27,3),ETA(28,28), ZETA(28,28)
              *.          AMASS(27,27), GAIRX(28,28), GAIRY(28,28), GWATX(27,27),
              *             GWATY(27,27), STRESS(28,28,3), DIV(28,28)
               DIMENSION DRAGS(28,28), DRAGA(28,28), HEFFM(28,28),
              * OUT(28,28),HEFF(28,28,3),AREA(28,28,3),COR(27,27),
              * DWATN(27,27)
               COMMON /TIME/ RELAXS,FORMS,ADVCTS,SMTHS,HEATS,MESHS,INITS
               COMMON /FORCE/ FORCEX(27,27),FORCEY(27,27)
               COMMON  /PRESS/  PRESS(28,28)
               COMMON /STEP/ DELTAT, DELTAX, DELTAY, DELTA1, DELTA
               DATA      FCOR /1.46E-04/
              *.          RHOAIR/1.3/
              *.          SINWIN/0.4226/
              *.          COSWIN/0.9063/
              *.          SINWAT/0.4226/
              *.          COSWAT/0.9063/
              *.          ECCEN/2.0/
               CALL SECOND(T1)
          C
          C      SET UP CORIOLIS TERM
          C
               DO 101 J = 1,27
               DO 101 I = 1,27
               AMASS(I,J) = 0.91E+03 * 0.25 * (HEFF(I,J,1) + HEFF(I+1,J,1) +
              *  HEFF(I,J+1,1) + HEFF(I+1,J+1,1))
               COR(I,J) = AMASS(I,J) * FCOR
               DWATN(I,J) = 5.5 * SQRT((UICE(I,J,1) - GWATX(I,J)) * * 2 +
              * (VICE(I,J,1) - GWATY(I,J)) * * 2)
               DRAGA(I,J) = DWATN(I,J) * SINWAT + COR(I,J)
           101 CONTINUE
          C
          C  SET UP NON LINEAR WIND & WATER DRAG
          C
               DO 105 J = 1,27
               JJ = J + 1
               DO 105 I = 1,27
               II = I + 1
               DAIRN = RHOAIR * .0012 * SQRT(GAIRX(II,JJ) * * 2 +
              * GAIRY(II,JJ) * * 2)
          C
          C** SET UP SYMMETRIC DRAG
          C
               DRAGS(I,J) = DWATN(I,J) * COSWAT
          C
          C** NOW SET UP FORCING FIELD
          C** FIRST DO WIND
               FORCEX(I,J) = DAIRN * (COSWIN * GAIRX(II,JJ) -
              * SINWIN * GAIRY(II,JJ))
               FORCEY(I,J) = DAIRN * (SINWIN * GAIRX(II,JJ) +
              * COSWIN * GAIRY(II,JJ))
          C
           105  CONTINUE
               DO 107 J = 1,27
               DO 107 I = 1,27
          C    NOW ADD IN CURRENT FORCE
```

```
TRAN 1.5.1 CYCLE FTN1586  BUILT 03/03/81 23.27   SOURCE LISTING
            SUBROUTINE PLAST(UICE,VICE,PRESS,ETA,ZETA,ECCEN,HEFFM,
     *    NX1,NY1,DIV)
C SUBROUTINE CALCULATES STRAIN RATES AND VISCOUS PARAMETERS
            DIMENSION UICE(27,27,3),VICE(27,27,3),PRESS(28,28)
     *,          ZETA(28,28), DIV(28,28)
     *,           STRESS(28,28,3), ETA(28,28)
     *,            HEFFM(28,28)
            DIMENSION E11(28,28), E22(28,28), E12(28,28)
            COMMON /STEP/ DELTAT, DELTAX, DELTAY, DELTA1, DELTA
            ECM2=1.0/(ECCEN**2)
             GMIN = 1.0E-20
C NOW EVALUATE STRAIN RATES
C
C  WE COULDNT FIND E11(1,1),E12(1,1),E22(1,1) UNTIL NOW
C  THEREFORE WE ASSIGN THEM EQUAL ZERO, THEY ARE COMPUTED
C  FROM VELOCITY AT THE BOUNDARIES *** THANS SPAT *
C
            E11      = 0.0
            E12      = 0.0
            E22      = 0.0
            ZMIN = 4.0E + 08
C*****
            DO 101 J = 2,27
            DO 101 I = 2,27
            E11(I,J) = (0.5/DELTAX) * (UICE(I,J,1) + UICE(I,J-1,1)
     *  -UICE(I-1,J,1)-UICE(I-1,J-1,1))
            E22(I,J) = (0.5/DELTAY) * (VICE(I,J,1) + VICE(I-1,J,1)
     *  -VICE(I,J-1,1)-VICE(I-1,J-1,1))
            E12(I,J) = (0.25/DELTAY) * (UICE(I,J,1) + UICE(I-1,J,1)
     *    -UICE(I,J-1,1)-UICE(I-1,J-1,1))
     *  +(0.25/DELTAX)*(VICE(I,J,1)+VICE(I,J-1,1)
     *  -VICE(I-1,J,1)-VICE(I-1,J-1,1))
  101 CONTINUE
C  NOW EVALUATE VISCOSITIES
C
            DO 110 J = 2,27
            DO 110 I = 2,27
            DELT = (E11(I,J) * * 2 + E22(I,J) * * 2) * (1.0 + ECM2) + 4.0 *
     *        ECM2 * E12(I,J) * * 2 + 2.0 * E11(I,J) * E22(I,J) *
     *        (1.0 - ECM2)
            DELT1=SQRT(DELT)
            DELT1=AMAX1(GMIN,DELT1)
            ZETA(I,J)=0.5*PRESS(I,J)/DELT1
C NOW PUT MIN AND MAX VISCOSITIES IN
  110 CONTINUE
            ZMIN = 4.0E+08
            DO 115 J = 1,27
            DO 115 I = 1,27
            ZMAX = (5.0E+12 / 2.0E+04) * PRESS(I,J)
            ZETA(I,J) = AMIN1(ZMAX,ZETA(I,J))
            ZETA(I,J) = AMAX1(ZMIN,ZETA(I,J))
  115 CONTINUE
            DO 120 J = 1,27
            DO 120 I = 1,27
            ETA(I,J)=ECM2*ZETA(I,J)
            E11(I,J) = E11(I,J) * HEFFM(I,J)
            E22(I,J) = E22(I,J) * HEFFM(I,J)
            E12(I,J) = E12(I,J) * HEFFM(I,J)
            SS11 = (ZETA(I,J) - ETA(I,J)) * (E11(I,J) + E22(I,J)) - PRESS(I
     *     * 0.5
```

```
TRAN 1.5.1 CYCLE FTN1536  BUILT 08/03/81 23.27   SOURCE LISTING        PLAST
              STRESS(I,J,1) = (2.0 * ETA(I,J) * E11(I,J) + SS11)
              STRESS(I,J,2) = 2.0 * ETA(I,J) * E22(I,J) + SS11
              STRESS(I,J,3) = 2.0 * ETA(I,J) * E12(I,J)
       C
       C      CALCULATE THE ICE DIVERGENCE AS THE SUM OF THE STRAIN RATES
       C
              DIV(I,J) = E11(I,J) + E22(I,J)
         120  CONTINUE
              RETURN
              END
    NO ERRORS
```

```
TRAN 1.5.1 CYCLE FTN1836  BUILT 09/03/81 23 27   SOURCE LISTING
               SUBROUTINE ADJUST(HEFF,AREA,OUT,HEFFM,NX,NY,NX1,NY1)
               DIMENSION HEFF(28,28,3),AREA(28,28,3)
               DIMENSION HEFFM(28,28), OUT(28,28)
               DIMENSION OUT2(28,28)
               CALL MEAN(HEFF,OUT2,NX,NY,OUT)
               DO 100 J = 2,27
               DO 100 I = 2,27
               HEFF(I,J,1) = HEFF(I,J,1) + (HEFFM(I,J) - OUT(I,J)) * OUT2(I,J)
         100   CONTINUE
               CALL MEAN(AREA,OUT2,NX,NY,OUT)
               DO 110 J = 2,27
               DO 110 I = 2,27
               AREA(I,J,1) = AREA(I,J,1) + (HEFFM(I,J) - OUT(I,J)) * OUT2(I,J)
         110   CONTINUE
               RETURN
               END
    NO ERRORS
```

```
FTN 1.5.1 CYCLE FTN1536  BUILT 04/03/81 23.27   SOURCE LISTING
              SUBROUTINE MESH(GRDI,GRDJ,NX,NY,NX1,NY1,N)
        CCC
        C     SUBROUTINE MESH
        C
        C     PURPOSE     TO CALCULATE THE FNOC I,J GRID POINTS FOR THE MODEL
        C                 GRID AND CALCULATE THE MESH SIZE.
        C
        C     USAGE
        C                 INPUT
        C                    READS      I0,I1      DEFINING I GRID POINTS
        C                               J0,J1      DEFINING J GRID POINTS
        C
        C                   I0,I1,J0,J1 DEFINED AS FOLLOWS
        C
        C                     X                           X
        C                   (I0,J1)                     (I1,J1)
        C
        C
        C
        C                     X                           X
        C                   (I0,J0)                     (I1,J0)
        C
        C                               N          NUMBER OF GRID POINTS ON A SIDE
        C
        C                 OUTPUT
        C                               GRDI,GRDJ   I,J GRID POINTS FOR THE MODEL
        C                                           GRID
        C                               DELTAX      MESH SIZE
        C
        C     METHOD
        C              THE DEFINING GRID POINTS ARE READ FROM THE INPUT STREAM.
        C     THESE VALUES ARE USED TO CALCULATE THE I,J POINTS OF THE MODEL
        C     GRID.  THE MAP FACTOR IS CALCULATED AT EACH POINT AND THE AVERAGE
        C     MAP FACTOR IS USED TO CALCULATE THE MESH SIZE OF THE GRID.
        C
        CCC
              IMPLICIT REAL ( A-Z )
              COMMON /BODY/ BODYI(20), BODYJ(20), BX(20), BY(20)
              COMMON /TIME/ RELAXS,FORMS,ADVCTS,GRWTHS,HEATS,MESHS,INITS
              DIMENSION GRDI(29,29), GRDJ(29,29)
              COMMON /STEP/ DELTAT,DELTAX,DELTAY,DELTA1,DELTA
             .INTEGER I,J,N,NX,NY,NX1,NY1
              INTEGER L,LL
              CALL SECOND(MESHS)
        C
        C     READ DEFINING POINTS
        C
              READ (7,10) I0,I1
              READ (7,10) J0,J1
              READ (7,12) N
          12  FORMAT(I5)
          10  FORMAT(2F10.2)
              PRINT 1000, I0,I1,J0,J1
         1000 FORMAT(1X,'DEFINING GRID POINTS',/1X,'I0 AND I1    ',2I5,
             *      /1X,'J0 AND J1    ',2I5)
              NX = N - 2
              NY= N - 2
              NX1 = N - 1
              NY1 = N - 1
        C
```

```
C     SET GRID INCREMENTS
C
C     SET UP J POINTS
C
      DELTA1 = (J1 - J0) / FLOAT(N-1)
      RI = J1
      DO 15 I = 1,N
      DO 13 J = 1,N
      GRDJ(I,J) = RI
   13 CONTINUE
      RI = RI - DELTA1
   15 CONTINUE
C
C
C
C     SET UP I MESH
C
      DELTA = (I1 - I0) / FLOAT(N-1)
      RJ = I0
      DO 25 I = 1,N
      DO 23 J = 1,N
      GRDI(J,I) = RJ
   23 CONTINUE
      RJ = RJ + DELTA
   25   CONTINUE
C
C
      DELTA = (DELTA + DELTA1) * 0.5
C
C     COMPUTE THE MESH SIZE
C
      SUM = 0.0
      DO 100 I = 1,N
      DO 100 J = 1,N
      RSQ = ((GRDI(I,J) - 31.0) * * 2) + ((GRDJ(I,J) - 31.0) * * 2)
      SINL = (1961.6426 - RSQ) / (1961.6426 + RSQ)
      XMAP = 1.86602540378 / (1 + SINL)
      SUM = SUM + XMAP
  100 CONTINUE
      XMAVG = SUM / (N * N)
      DELTAX = DELTA * 381000.0 * XMAVG
      DELTAX = AINT(DELTAX)
      DELTAY = DELTAX
      PRINT 1005,XMAVG,DELTAX
 1005 FORMAT(1X,*AVERAGE MAP FACTOR IS  *,F10.3,
     *       /1X,*THE MESH SIZE IS (IN METERS)   *,F10.3)
C
C     COMPUTE LOCAL GRID POINTS OF BUOYS
C
      D = 1.0 / DELTA
      D1 = 1.0 / DELTA1
      DO 135 L = 1,20
      I1 = BUOYI(L) - I0
      J1 = BUOYJ(L) - J0
      BX(L) = I1 * D
      BY(L) = J1 * D1
  135 CONTINUE
      PRINT 77
      PRINT 77,(BX(LL),LL=1,20)
   77 FORMAT(1H0,*LOCAL BUOY GRID POINTS  */1X,20F4.1) ·
```

```
TRAN 1.5.1 CYCLE FITNESS  BUILT 08/03/81 23 27   SOURCE LISTING        VERS
                 PRINT 78,(RY(LL),LL=1,20)
          78     FORMAT(1H0,'  Y POINTS  '/1X,20F4.1)
                 RETURN
                 END
   NO ERRORS
```

```
TRAN 1.5.1 CYCLE FTN1564  BUILT 13/03/81 23.27   SOURCE LISTING
              SUBROUTINE DIVERG(DIV,NX,NY,GRDI,GRDJ,IDTG,ITAU)
              DIMENSION  DIV(24,24)
              DIMENSION GRDI(24,23), GRDJ(23,24), IDTG(3), ITITL(6)
        C
        C
        C
              COMMON /DV/ MDV(24), DIVRG(625), FILLV(375)
              CHARACTER*8 MDV,LABEL(2),IDTG,ID1
              DATA ID1    /8HDIV,  24/
             *     MDV(3) /8H     64/
             *,    MDV(4) /8HFP,6400UY/
        C
        C     OUTPUT THE DIVERGENCE FIELD
        C
              K = 0
              DO 10 J = 2,NY
              DO 10 I = 2,NX
              K = K + 1
              DIVRG(K) = DIV(I,J)
           10 CONTINUE
              TITLE = 7HVASENVO
              LABEL(1) = ID1
              MDV(1) = ID1
              MDV(2) = IDTG(1)
              LABEL(2) = MDV(2)
              CALL PORTFF(TITLE,LABEL,MDV,644,ISTAT)
              IF(ISTAT .NE. 0) GO TO 1000
              RETURN
         1000 PRINT 1010,ISTAT,LABEL
         1010 FORMAT(1H0,'CWRITE STATUS IS ',I5,' ON WRITE OF ',A8)
              STOP 'BAD WRITE IN DIVERG'
              END
    NO ERRORS
```

```
TRAN 1.5.1 CYCLE FTN1F36  BUILT 08/03/81 23 27    SOURCE LISTING
                  SUBROUTINE INITIAL(NO,GATR,GRDI,GRDJ,IDTS,N,ITAU)
                  DIMENSION GRDI(24,22), GRDJ(22,24), GATR(24,24), LABEL(2),
                 *               IPCD(8)
                  COMMON /TIME/ RELAXS,FORCS,ADVCTS,GRWTHS,HEATS,MESHS,INITS
                  LOGICAL IP(8), MASK1,MASK2,IDL,ITL
                  LOGICAL MASK3,MASK4
                  COMMON /BUFFER/ IDENT(24), DATA(63,63), FILL(107)
                  CHARACTER*8 IDENT,IT,IDTS(3),LABEL,IPCD
                  EQUIVALENCE (IT,ITL), (IP,IPCD), (IDL,IDENT(1))
                  DATA IDENT(3) /8H     3343/
                  DATA IDENT(8) /8H     3333/
                  DATA MASK1 /X'FFFFFFF0000000000001'/
                 *,      MASK2 /X'00000002420000000001'/
                 *,      MASK3/X'00000000000000000FFF'/
                 *,      MASK4/X'00000000000000202000'/
                 *,      (IPCD(KK),KK=1,8) /3HA20, 3HA21, 3HA10, 3HA01, 3HA12,
                 *                                    3HA11, 3HA18, 3HA15/
                  IFILE = 5HTAPE9
                  CALL SECOND(T1)
                  CALL BINASC(ITAU,IT)
                  ITL = ITL .AND. MASK3
                  ITL = ITL .OR. MASK4
                  IDL = IP(NO) .AND. MASK1
                  IDL = IDL .OR. MASK2
                  IDL = IDL .OR. ITL
                  IDENT(2) = IDTS(1)
                  LABEL(2) = IDENT(2)
                  LABEL(1) = IDENT(1)
                  PRINT 600,LABEL(1),LABEL(2)
        600       FORMAT(1X,'INITIAL       READING RECORD    ',2A8)
                  CALL CHECKNC(IFILE,LABEL,3443,LEN,IS)
                  IF(IS .EQ. 0) STOP 'CHECKNC NO DATA INITIAL'
                  CALL READER(IFILE,LABEL,IDENT,3443,LEN,IS)
                  DO 200 I = 1,N
                  DO 200 J = 1,N
                  CALL INTRP(DATA,63,63,GRDI(I,J),GRDJ(I,J),GATR(I,J))
        200       CONTINUE
                  DO 500 J = 27,35
                  PRINT 75,(DATA(I,J),I=25,35)
        500       CONTINUE
        75        FORMAT(1X,11F10.4)
                  CALL SECOND(T2)
                  INITS = INITS + (T2 - T1)
                  RETURN
                  END
     NO ERRORS
```

```
RTRAN 1.5.1 CYCLE FTN1535  BUILT 06/03/81 23 27   SOURCE LISTING
               SUBROUTINE HAPLOT(HEFF,AREA,IDTG,ITAU,GRDI,GRDJ,NX,NY)
        CCC
        C      SUBROUTINE HAPLOT
        C
        C      PURPOSE   TO FORMAT THE THICKNESS AND COMPACTNESS FOR PRINTED AND
        C                PLOTTED OUTPUT.
        C
        C      USAGE
        C                INPUT
        C                      HEFF        ICE THICKNESS
        C                      AREA        ICE COMPACTNESS
        C                      PRESS       ICE PRESSURE OR ICE STRENGTH
        C                      IDTG        DATE TIME GROUP
        C                   GRDI,GRDJ      I, J POINTS
        C                      NX,NY       GRID DIMENSIONS
        C
        C      COMMON BLOCKS
        C               /PLOT/, /COEF/, /DATAI/, /ZRAND/
        C
        C      EXTERNALS
        C               DAY, EXTRAP, IDENTC, TABLE, ZDELFI, ZWRITI, ZILCH
        C
        C      METHOD
        C               THE THICKNESS VALUES ARE CONVERTED TO CM AND ARE WRITTEN
        C      TO TH ZRANDIO DATA FILE. THE COMPACTNESS VALUES ARE WRITTEN TO
        C      THE ZRANDIO FILE ALSO. THE OLD RECORDS ARE DELETED. BOTH OF THE
        C      ARRAYS ARE SENT TO TABLE FOR PRINTED OUTPUT.
        C               THE PLOT FILES ARE PREPARED BY FIRST SETTING UP THE 20
        C      WORD IDENTIFICATION BLOCK NECESSARY FOR FNOC DATA RECORDS.
        C      THE DATA VALUES ARE THEN PUT IN TERMS OF THE 63 - 63 GRID
        C      AND WRITTEN TO THE ZRANDIO DATA FILE FOR PLOTTING.
        C
        CCC
              COMMON /AP/ ID(24), ARRAY(625), FL(375)
              COMMON /PRESS/ PRESS(24,24)
              DIMENSION HEFF(24,24,3), AREA(24,24,3)
              DIMENSION GRDI(24,24), GRDJ(24,24)
              DIMENSION IDTG(3),IRCD(3)
              CHARACTER*8 ID, IDTG, IT, LABEL(2)
              CHARACTER*8 ID0
              CHARACTER*8  ID1,ID2
              DATA ID0/8HTHK,  24/
             *,    ID(3) /8H      649/
             *,    ID1    /8HCOM,  24/
             *,   ID2    /8HPRS,  24/
             *,    ID(4) /8HEP640DAY/
               ID(2) = IDTG(1)
               CALL PRIMASC(ITAU,IT)
        C
        C       WRITE OUT THE THICKNESS AND COMPACTNESS DATA FIELDS
        C
               K = 0
              NNX = NX - 1
              NNY = NY - 1
              DO 10 I = 2,NNX
              DO 10 J = 2,NNY
               K = K + 1
               ARRAY(K) = HEFF(I,J,1) * 100.0
           10  CONTINUE
              IFILE = 7HMASENKO
```

```
FTRAN 1.5.1 CYCLE FTN1E06  BUILT 08/03/81 23 27   SOURCE LISTING          IVPLOT    2
          C     INVERSE . X AND Y  MUST BE EXPRESSED IN GRID DISTANCE
          C     FROM THE POLE ./
          CCCC
                CALL INDDFF(ARRAYX(I,J),ARRAYY(I,J),DD(K),FF(K),X,Y)
          C
          C     CONVERT THE MOVEMENT FROM METERS/SEC. TO KNOTS
          C
                FF(K) = (FF(K) / 0.5144) * 100.0
            100 CONTINUE
                TITLE = 7HMASENVC
                LABEL(1) = ID(1)
                LABEL(2) = ID(2)
                CALL CWRITER(TITLE,LABEL,ID,642,ISTAT)
                IF(ISTAT .NE. 0) GO TO 1000
                IE(2) = ID(2)
                LABEL(1) = IE(1)
                CALL CWRITER(TITLE,LABEL,IE,642,ISTAT)
                IF(ISTAT .NE. 0) GO TO 1000
                RETURN
           1000 PRINT 1010,ISTAT,LABEL(1)
           1010 FORMAT(1X,'STATUS IS   ',I5,' ON WRITE OF  ',A6)
                STOP 'BAD WRITE IN IVPLOT'
                END
      NO ERRORS
```

```
TRAN 1.5.1 CYCLE FTN1586  BUILT 08/03/81 23.27   SOURCE LISTING
              SUBROUTINE UVPLOT( ARRAYX,ARRAYY,IDTG,NX,NY,GRDI,
             *                   GRDJ,ITAU)
CCC
C     SUBROUTINE UVPLOT
C
C     PURPOSE              TO OUTPUT THE DIRECTION,SPEED VECTORS OF ICE DRIFT
C                          FOR PRINTED AND PLOTTED OUTPUT.
C
C     USAGE
C            INPUT    ARRAYX          ARRAY OF U ICE DRIFT COMPONENT
C                     ARRAYY          ARRAY OF V ICE DRIFT COMPONENT
C                     IDTG            DATE TIME GROUP
C                     NX,NY           GRID DIMENSIONS
C                     ITAU            FORECAST TAU VALUE
C
C     COMMON BLOCKS
C           /PLOT/, /DDFF/, /ZRAND/
C
C     EXTERNALS
C           DAY, TABLE, UVDDFF, ZDELEI, ZWRIEI
C
C     METHOD
C           THE U,V, COMPONENTS ARE CONVERTED TO A METEOROLOGICAL
C     DIRECTION AND SPEED FOR THE I,J GRID. THE SPEED IS CONVERTED TO
C     KNOTS AND MULTIPLIED BY 100. THESE VALUES(DIRECTION AND FORCE)
C     ARE WRITTEN TO THE ZRANDIO FILE AND THE OLDER RECORDS ARE DELETED.
C           THE DIRECTION AND FORCE ARE THEN PACKED INTO ONE WORD,
C     TOGETHER WITH THE RESPECTIVE I,J GRID VALUE. THIS RECORD IS THEN
C     WRITTEN TO THE ZRANDIO DATA BASE.
C           SUBROUTINE TABLE IS CALLED TO FORMAT THE DIRECTION AND
C     FORCE FOR PRINTED OUTPUT.
C
CCC
      COMMON /DD/ ID(24),DD(625),FILLD(375)
      COMMON /FF/ IE(24),FF(625),FILLF(375)
      DIMENSION ARRAYX(27,27), ARRAYY(27,27)
     *,           IDTG(3), GRDI(29,29), GRDJ(29,29)
      CHARACTER*8 IPRD(2),IT,ID,IE,LABEL(2),IDTG
      DATA  IE(1) /8HFFFF,  24/
     *,       ID(1) /8HDDDD,  24/
     *,     ID(3)/8H       649/
     *,     IE(3)/8H       649/
     *,     ID(4) /8HEP6400JY/
     *,     IE(4) /8HEP6400JY/
      CALL 8HINASC(ITAU,IT)
      ID(2) = IDTG(1)
      K = 0
      NNX = NX - 1
      NNY = NY - 1
      DO 100 I = 2,NNX
      DO 100 J = 2,NNX
      K = K + 1
      II = I + 1
      JJ = J + 1
      X = GRDI(II,JJ) - 31.0
      Y = GRDJ(II,JJ) - 31.0
CCC
C     CALL UVDDFF( U,V,DD,FF,X,Y ) IS USED TO CONVERT GRID
C     ORIENTED WIND COMPONENTS( U,V ) ON POLAR STEREOGRAPHIC
C     GRID TO DIRECTION ( DD ) AND MAGNITUDE ( FF ) ,AND THE
```

```
RTRAN 1.5.1 CYCLE FTN1536  BUILT 08/03/81 23 27    SOURCE LISTING
                SUBROUTINE STATPRT(TSTOP)
                COMMON /TIME/ RELAXS,FORMS,ADVCTS,GRWTHS,HEATS,MESHS,INITS
                PRINT 1
          1     FORMAT('1','    T I M E   S T A T I S T I C S')
                PRINT 10,TSTOP
         10     FORMAT(1X,'TOTAL ICE MODEL TIME               ',F10.4)
                PRINT 20,RELAXS
         20     FORMAT(1X,'RELAXATION TIME . . . . . . . ',F10.4)
                PRINT 30,FORMS
         30     FORMAT(1X,'FORMS AND PLASTIC TIME    . . ',F10.4)
                PRINT 40,ADVCTS
         40     FORMAT(1X,'ADVECTION TIME    . . . . . . ',F10.4)
                PRINT 50,GRWTHS
         50     FORMAT(1X,'GROWTH TIME . . . . . . . . . ',F10.4)
                PRINT 60,HEATS
         60     FORMAT(1X,'HEAT BUDGET TIME    . . . . . ',F10.4)
                PRINT 70,MESHS
         70     FORMAT(1X,'INITIALIZING GRID AND OCEAN ',F10.4)
                PRINT 80,INITS
         80     FORMAT(1X,'READING GRADDIO TIME    . . . ',F10.4)
                RETURN
                END
    NO ERRORS
```

```
RTRAN 1.5.1 CYCLE FTN1536 BUILT 08/03/81 23 27   SOURCE LISTING                    6
                SUBROUTINE PRNT(ARRAY,I,J,K,M1,M2,N)
          CCC
          C     SUBROUTINE PRNT
          C
          C     PURPOSE:      TO PRINT A MODEL ARRAY
          C
          C     USAGE
          C                   ARRAY      THE ARRAY TO BE PRINTED
          C                   I,J,K      DIMENSIONS OF ARRAY
          C                   M,N        ROWS AND COLUMNS OF ARRAY TO BE PRINTED
          C
          CCC
                DIMENSION ARRAY(I,J,K)
                PRINT 5
                PRINT 7,M1,M2
            7   FORMAT(1X,'J FROM  ',I3,' TO ',I3)
                DO 10 KK = 1,I
                DO 10 II = 1,N
                PRINT 20,(ARRAY(II,JJ,KK),JJ=M1,M2)
           10   CONTINUE
                PRINT 5
            5   FORMAT(///)
           20   FORMAT(1X,13F9.3)
                RETURN
                END
      NO ERRORS
```

```
            IDA(1) = ID0
            LABEL(1) = IDA(1)
            LABEL(2) = IDA(2)
            CALL WRITTER(IFILE,LABEL,IDA,649,ISTAT)
            IF(ISTAT .NE. 0) GO TO 1000
      C
      C       C O M P A C T N E S S
      C
              K = 0
            DO 20 I = 2,NNX
            DO 20 J = 2,NNY
              K = K + 1
              ARRAY(K) = AREA(I,J,1)
   20         CONTINUE
            IDA(1) = ID1
            LABEL(1) = IDA(1)
            CALL WRITTER(IFILE,LABEL,IDA,649,ISTAT)
            IF(ISTAT .NE. 0) GO TO 1000
      C
      C     ICE PRESSURE
      C
               K = 0
            DO 30 I = 2,NNX
            DO 30 J = 2,NNY
               K  =  K + 1
            ARRAY(K) = PRESS(I,J)
   30       CONTINUE
            IDA(1) = ID2
            LABEL(1) = IDA(1)
            CALL WRITTER(IFILE,LABEL,IDA,649,ISTAT)
            IF(ISTAT .NE. 0) GO TO 1000
            RETURN
 1000       PRINT 1010,ISTAT,LABEL
 1010       FORMAT(1H0,*WRITE STATUS IS  *,I5,*  ON WRITE OF  *,A10)
            STOP *BAD WRITE IN HAPLOT*
            END
    NO ERRORS
```

```
FORTRAN 1.5.1 CYCLE FTN15G6  BUILT 08/03/81 23 27   SOURCE LISTING       INTRP
C
C SIMPLIFIED FORMULA:
 121  GI2=XI2+XI2+XI2-XI3-XI3
      GJ3 = YJ3 - YJ2
      GJ4 = YJ - YJ2 + GJ3
      GJ2 = YJ2 - GJ3 - GJ3
      GJ1 = 1.0 - GJ2
      PHI =(P4*GJ1+P8*GJ2+((P11-P4)*GJ3+(P8-P1)*GJ4)*0.5)*(1.-GI2)
     +    +(P5*GJ1+P9*GJ2+((P12-P5)*GJ3+(P9-P2)*GJ4)*0.5)*GI2
     +    +0.5*( ((P6-P4)*GJ1+(P10-P4)*GJ2)*(XI3-XI2)
     +          +((P5-P3)*GJ1+(P9- P7)*GJ2)*(XI-XI2-XI2+XI3) )
      RETURN
C
C
C     SPECIAL CASES:
C
C SPECIAL CASE 1: YJ=0
 51   P5=Z(L+1)
      IF (I.NE.1) GO TO 61
      P3 = P4 + P4 - P5
      GO TO 62
 61   P3=Z(L-1)
 62   IF (I.NE.(M-1)) GO TO 63
      P6 = P5 + P5 - P4
      GO TO 64
 63   P6 = Z(L + 2)
      GO TO 64
C
C SPECIAL CASE 2: XI=0
  52  P5 = Z(L + M)
      XI = YJ
      IF (J.NE.1) GO TO 81
      P3 = P4 +P4 - P5
      GO TO 82
 81   P3=Z(L-M)
 82   IF (J.NE.(N-1)) GO TO 83
      P6 = P5 + P5 - P4
      GO TO 64
 83   P6 =Z(L+M+M)
C
C SPECIAL FORMULA:
 64   XI2 = XI * XI
      XI3 = XI2 * XI
      GI3=XI3-XI2
      GI2=XI2-GI3-GI3
      GI4=XI-XI2+GI3
      PHI=(1.-GI2)*P4+GI2*P5+(GI3*(P6-P4)+GI4*(P5-P3))*0.5
      RETURN
C
C
C SPECIAL CASE 3: XI=YJ=0
 53   PHI=P4
      RETURN
C
C
C POINT Z(X,Y) OUT OF BOUNDS:
 99   PHI=MASK(59)
      RETURN
C
      END
    NO ERRORS
```

```
ORTRAN 1.5.1 CYCLE FTN1586  BUILT 08/03/81 23 27   SOURCE LISTING
          SUBROUTINE GROWDEC(HDIFF,FHEFF,GAREA,IDTS,ITAJ,NX1,NY1)
          DIMENSION HDIFF(24,24), FHEFF(24,24), GAREA(24,24)
          COMMON /DV/ MDV(24), DIVRG(625), FILLV(375)
          CHARACTER*4 MDV, LABEL(2), IDTS(3), ID, ID1
          CHARACTER*8 ID0
          DATA  MDV(3) /8H      649/
         *,   ID0 /8HHDF+  24/
         *,      ID     /8HFHEF+  24/
         *,      ID1    /8HGAR+  24/
         *,    MDV(4) /8HEP6400UY/
          MDV(2) = IDTS(1)
          IFILE = 7HMASENWC
C
C         OUTPUT THE OPEN WATER GROWTH
C
          K = 0
          NNX = NX1 - 1
          NNY = NY1 - 1
          DO 10 I = 2,NNX
          DO 10 J = 2,NNY
          K = K + 1
          DIVRG(K) = HDIFF(I,J)
    10    CONTINUE
          MDV(1) = ID0
          LABEL(1) = MDV(1)
          LABEL(2) = MDV(2)
          CALL CWRITER(IFILE,LABEL,MDV,649,ISTAT)
          IF(ISTAT .NE. 0) GO TO 1000
C
C         OUTPUT THE SH TERM OF THE THERMO. CONTINUITY EQUATION
C
          K = 0
          DO 20 I = 2,NNX
          DO 20 J = 2,NNY
          K = K + 1
          DIVRG(K) = FHEFF(I,J)
    20    CONTINUE
          LABEL(1) = ID
          MDV(1) = ID
          CALL CWRITER(IFILE,LABEL,MDV,649,ISTAT)
          IF(ISTAT .NE. 0) GO TO 1000
C
C         OUTPUT THE SA TERM OF THE THERMO CONTINUITY EQUATION
C
          K = 0
          DO 30 I = 2,NNX
          DO 30 J = 2,NNY
          K = K + 1
          DIVRG(K) = GAREA(I,J)
    30    CONTINUE
          LABEL(1) = ID1
          MDV(1) = ID1
          CALL CWRITER(IFILE,LABEL,MDV,649,ISTAT)
          IF(ISTAT .NE. 0) GO TO 1000
          RETURN
  1000    PRINT 1010,ISTAT,LABEL
   1010 FORMAT(1H0,'CWRITE STATUS IS  ',I5,'  ON WRITE OF  ',2A8)
          STOP 'BAD WRITE IN GROWDEC'
          END
    NO ERRORS
```

```
FORTRAN 1.5.1 CYCLE FTN1536  BUILT 09/03/81 23.27   SOURCE LISTING
                SUBROUTINE BUOYD(UICEC,VICEC,GRDI,GRDJ,NX,NY)
                COMMON /STEP/ DELTAT,DELTAX,DELTAY,DELTA1,DELTA
                COMMON /BUOY/ BUOYI(20),BUOYJ(20),BX(20),BY(20)
                DIMENSION UICEC(27,27), VICEC(27,27), GRDI(24,24), GRDJ(24,24)
                PRINT 100
                DO 50 L = 1,20
                IF(BX(L) .LT. 1) GO TO 1000
                IF(BX(L) .GT. NX) GO TO 1000
                IF(BY(L) .LT. 1) GO TO 1000
                IF(BY(L) .GT. NY) GO TO 1000
                CALL INTPP(UICEC,NX,NY,BX(L),BY(L),U)
                CALL INTPP(VICEC,NX,NY,BX(L),BY(L),V)
                U = (U * 86400.0) / DELTAX
                V = (V * 86400.0) / DELTAY
                SPACEI = U * DELTA
                SPACEJ = V * DELTA1
                BUOYI(L) = BUOYI(L) + SPACEI
                BUOYJ(L) = BUOYJ(L) + SPACEJ
                BX(L) = BX(L) + U
                BY(L) = BY(L) + V
                PRINT 105
                PRINT 105,L,BX(L),BY(L),BUOYI(L),BUOYJ(L)
          105   FORMAT(1X,I5,4F10.4)
          100   FORMAT(1H0,'BUOY NO.  ','    X      ','    Y      ','    I     ',
               *    '    J     ')
          1000 CONTINUE
          50    CONTINUE
                RETURN
                END
     NO ERRORS
```

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER
NORDA Tech Note 122

2. GOVT ACCESSION NO.
AD-A114 343

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)
Program Maintenance Manual Polar Ice Forecast Subsystem - Arctic

5. TYPE OF REPORT & PERIOD COVERED
Preliminary FY81

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)
P. A. Harr, T. C. Pham and J. P. Welsh

8. CONTRACT OR GRANT NUMBER(s)
N0014-81-F-0028

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Control Data Corporation
205 Montecito Ave.
Monterey, CA 93940

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
93321D work unit

11. CONTROLLING OFFICE NAME AND ADDRESS
Naval Air System Command
Jefferson Davis Highway
Arlington, VA 22210

12. REPORT DATE
October 1981

13. NUMBER OF PAGES
146

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)
Polar Oceanography Branch
NORDA
NSTL Station, MS 39529

15. SECURITY CLASS. (of this report)
UNCLASSIFIED

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
None

16. DISTRIBUTION STATEMENT (of this Report)

Unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

Unlimited

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Numerical model	Sea Ice
Polar	Arctic
Ice Forecasting	Program maintenance

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This manual provides a complete and detailed description of the dynamic thermodynamic sea ice model (PIFS-N) including all the necessary information for maintenance programmer personnel required to maintain the system.

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

ILMED

-8